Study on Algorithm of Contours Path Generation for Robotic Prototyping

Zhao Jibin  Liu Weijun  Wang Yuechao
Advanced Manufacture Technology Lab, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, Liaoning 110016, PR China.

Abstract—This paper introduces some key technologies in contours path generation. Slicing result of STL (Stereolithography) file is a set of closed polygons of cross-sectional contours. The cross-sectional contours must be offset a distance inward solid area in the course of machining parts because of the size of laser light spot. A simple calculating method of offset contours is presented. Aiming at self-intersection of offset contour, a modified algorithm of detecting self-intersection point is proposed. The algorithm reduced the time complexity. To eliminate the invalid loops, a uniform method to identify invalid loops is proposed and an effective removing algorithm of invalid loops is presented. The algorithms of this paper not only improve the efficiency of program but also enhance the precision of part.

Keywords—robotic prototyping; path generation; CAD/CAM

I. INTRODUCTION

Rapid Prototyping(RP) is a key Technology of the 1990s. More than a dozen RP technologies have emerged since the first RP technology, stereolithography, was commercialized in 1988. In contrast to traditional material removal processes, the RP technologies build a part by gradually adding material layer-by-layer[1]. A CAD model can be directly transform into product model by RP technology. Robot is a type of perfect automatic equipment. Robotic technology used in rapid prototyping is called robotic prototyping system. A robotic prototyping system is used instead of traditional CNC devices due to the following reasons: With the same size of workshop, the robot prototyping system has a 40% of floor space reduction over ordinary CNC devices.

For rapid prototyping processing, the STL models need to be sliced into a set of X-Y sectional contours before it can be used to build a RP object; the sectional contours consist of a set of polygon. But the cross-sectional contours only are the theoretical contours. Because of the size of laser light spot, the contours must be offset a laser radius to keep the precision of parts machined. The laser path compensation of section contours in RP system is the polygons offsetting in fact. Polygon offsetting is a fundamental operation in computer graphics, the operations play important role in CAD/CAM and robot path planning. Offset contours of some polygons can generate self-intersection, come into being the self-loop. Fig.1 shows an example of self-loop of offsetting contour. The exact result is the offsetting contours after eliminating invalid self-loops (see Fig.1(b)). But eliminating all invalid loops is very difficult, because the intersection-detection is time consuming, and the loop-removal is prone to numerical errors. The voronoi diagram method is known to be more efficient and robust[2,3], but it may also suffer from numerical instability. Kalmanovich and Nisnevich[4] proposed an algorithm for removing invalid loops still suffer same problem. Liu[5] has studied the laser path compensation of LOM (Laminate Objective Manufacturing), and Zhao[6] has studied the problem of stereolithography. But they all did not deal with the loop problem. Sometimes, the shape of part is very intricate, especially the CAD model of part gained by reverse engineering. The parts have lots of delicate structure at some portions. In the case, dispose to the loop problem of section contours offsetting is indispensable. Starting in 1986 with the stereolithography of polymers, the resolution and precision have been improved down to a few micrometers[7]. If the invalid loop is not removed, the part to be manufactured could be failure.

Figure 1 invalid loop and its removal

In this paper, some key technologies are introduced in laser path compensation in stereolithography. Certainly, all algorithms are also suit for FDM. Firstly, a simple cross-sectional contours offset algorithm is present. Secondly, a modified algorithm of self-intersection point detection is proposed. The algorithm reduced the time complexity. Thirdly, we introduced a uniform method to identify
invalid loops. An effective removing algorithm of invalid loops is presented.

II. CROSS-SECTIONAL CONTOURS OFFSET ALGORITHM

A. Offset direction decision

Before a part is fabricated in a RP system layer-by-layer, the STL model of the part needs to be sliced to obtain cross-sectional contours. Noting that the cross-sectional contours consist of one or several closed polygons. According to rule of geometry, clockwise is the positive direction and counter clockwise is negative direction for the outer outline of solid. In this way, counter clockwise is positive direction and clockwise is negative for the inner outline of solid. STL file format is a polyhedral representation of the part with triangular facets. Because information of each triangular facet in STL file includes outer normal vector, direction of cross-sectional contours can be decided in the course of slicing. In this case, the solid area always lies in left side, when we go along the positive direction of contours. In this paper, offset algorithm of stereolithography only is discussed. In the case, the contours offset left side along the positive direction, namely the contours offset inward solid area.

B. Calculation method of offset contours

Vertices calculating method of offsetting contours is shown in Fig.2. Without loss of the generality, set the origin of the temporary coordinate system at the arbitrary vertices of sectional contours \( P_i, L_1, L_2 \) and \( L_2' \) are the vectors formed by \( P_i, P_1, P_2 \), and \( P_i P_{i+1} \). Offset distance \( R \) is laser light spot radius, and \( \alpha \) is the internal angle between vector \( P_i P_1, \) and \( P_i P_{i+1} \), then the unit vector of \( L_1 \) and \( L_2 \) is written as

\[
\vec{L}_1 = x_1 i + y_1 j \\
\vec{L}_2 = x_2 i + y_2 j
\]

Let \( L_1' \) and \( L_2' \) be the equidistant line of \( L_1 \) and \( L_2 \), then we get the following equations.

\[
\begin{align*}
x_1 y - y_1 x &= R \\
x_2 y - y_2 x &= R
\end{align*}
\]

(1)

Since \( L_1 \) and \( L_2 \) are not parallel to each other, solve equation (1) to get

\[
\begin{align*}
x_p = (x_2 - x_1) R / (x_1 y_2 - x_2 y_1) \\
y_p = (y_2 - y_1) R / (x_1 y_2 - x_2 y_1)
\end{align*}
\]

(2)

Thus we can obtain the coordinate values of all the offset vertices.

As shown in Fig.3, when internal angle is greater than \( 1.5\pi \), laser path generates a sharp vertex [8] in terms of formula (3). The distance between vertex \( p' \) and \( p \) is so far that laser travel unnecessary path. To reduce unnecessary laser travel, we need to employ different method for a vertex having an internal angle larger than \( 1.5\pi \). In this paper, two vertex is employed instead the sharp vertex, as shown in Fig.3. a vertex is added according to in Fig.3. However, we would prefer the laser path, because its length is shorter. Both vertex are expressed as

\[
\begin{align*}
x_{T_1} &= (x_1 - y_1) R \\
y_{T_1} &= (x_1 + y_1) R
\end{align*}
\]

(3)

\[
\begin{align*}
x_{T_2} &= -(x_2 + y_2) R \\
y_{T_2} &= (x_2 - y_2) R
\end{align*}
\]

(4)

Figure 3. Calculation method of \( \alpha > 1.5\pi \)

III. DETECTING ALGORITHM OF SELF-INTERSECTION POINT

Every offset section also consists of one or several polygons like the cross-sectional contour. Laser path is the offset contours which self-intersection and loops is removed. Invalid loops are removed to ensure laser path is simple and closed. To eliminate the invalid loops, we first need to detect the self-intersections among the offset contour.

The present algorithm in the paper is the modified Bentley-Ottmann’s sweep-line algorithm[9]. Because each monotone chain has no self-intersection point, the present algorithm detects the self-intersection points between two monotone chains without splitting the offset polygon[10] in optimal sweeping direction.

A. Determination of sweeping direction

The number of monotone chain in a polygon depends on the sweeping direction, so an optimal sweeping direction can minimize the number of monotone chain. As shown in Fig.5, the polygonal chain consists of four monotone chains in x-axis direction, while the polygonal chain consist of ten monotone chains. Therefore, minimizing the number of monotone chain can optimize the time complexity of intersection detection algorithm.
In common case, the sweeping direction is selected in x-axis and y-axis. Sweeping direction is determined by the number of extreme points of x direction and y direction. If the number of extreme points of x direction (y direction) less than that of y direction (x direction), x direction (y direction) is regarded as sweeping direction.

### B. Intersection calculation

An intersection can be detected by checking whether there exists any intersection current monotone chain and other monotone because there does not exit intersection in one monotone chain.

Fig.4 shows a polygonal chain with an intersection. The intersection point \( I^* \) between segments \( P_iP_{i+1} \) and \( P_jP_{k+1} \) (see Fig.4) can be calculated as follows

\[
I^* = P_j + tP_iP_{i+1} = P_k + sP_jP_{k+1}
\]

\[
0 \leq t \leq 1.0 \text{ and } 0 \leq s \leq 1.0
\]

By solving the Eq.5, we can find the parameters \( t \) and \( s \) as follows

\[
t = \frac{P_kP_j \times P_{i+1}P_{i+1}}{P_{i+1}P_{i+1} \times P_kP_{k+1}}
\]

\[
s = \frac{P_jP_k \times P_{i+1}P_{i+1}}{P_{i+1}P_{i+1} \times P_jP_{k+1}}
\]

if \( P_iP_{i+1} \) and \( P_jP_{k+1} \) intersect each other, both the parameters \( t \) and \( s \) have to meet

\[
0 \leq s \leq 1.0
\]

\[
0 \leq t \leq 1.0
\]

The intersection point \( I^* \) can be calculated by substituting \( s \) or \( t \) in Eq.5.

![Figure 4 Calculating method of self-intersection](image)

### C. Intersection point detection algorithm

To detect intersection efficiently, we present four important data structure to be used in the intersection detection algorithm.

1. Offset polygonal contour list (OutlineList) is a list of sequence of vertex, and it is a closed polygon. It contains data of vertex with doubly linked list. The number of vertex of the offset polygon is stored in head node. Storing sequence of vertex is clockwise for outer contours and anti-clockwise for inner contours. Fig.5 shows OutlineList \( = \{V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9, V_10, V_{11}, V_{12}\} \).

2. Active edge list is a list of active edge being crossed by the sweep line. Initially, the active edge list is empty. As the algorithm progress, the edge its front vertex located right after the sweep line is designated as the active edge and the new edge is added; while the edge its front vertex located right before the sweep line removed from the active edge list. Each edge in active edge list respectively belongs to different monotone chain.

3. Sweeping chain list is a list of edges that have not been completely processed yet. The edges are ordered by the front vertex of the edge in an increasing order of x value (y value).

4. Intersection point list contains all intersection point and their start point of edges intersected found while sweeping a polygon from left to right (down to up).

![Figure 5 An polygon with self-intersection](image)
Step5: In the light of Eq.6 and Eq.7, detect the intersection between the active edge and the inactive edges. If have intersection, go to Step6; otherwise, go to Step 3.

Step6: In light of Eq.5, calculate the intersection point; insert the intersection point and the start points of both edges intersected into Intersection point list.

Step7: Repeat Step3-Step6, until all edges of Sweeping chain list is visited.

IV. PROCESSING LOOPS OF AN OFFSET POLYGON

To eliminate the loops of offset polygon, we need to determine which loops to be removed (see Fig 6). There are several loop removal methods in the literature[11,12]: (1) consider the number of segments in each loop and remove the loop with fewest segments; (2) consider the direction of loops and remove the loops with different direction according to origin direction; (3) consider loop nesting and remove the loop that is insides another loop. These methods not work sometimes or computing method is complicated. In the section, we present a simple method to remove effectively the invalid loops.

Since the offset polygons also have direction as polygonal contours, namely the outer contours is clockwise and inner contour is anticlockwise. An invalid loop is confirmed by calculating the vector product at intersection point(Eq.10).

\[ F = P_k \times P_{k+1} \]

(10)

Therefore, the loops are identified by Eq.10. If \( F < 0 \), the loop is invalid; if \( F > 0 \), the loop is valid.

The method also works for the nesting loop. As shown in Fig.6, the polygonal contour is inner contour and its direction is anticlockwise. The polygonal contour has a concave portion result in nesting problem occurs. Applying Eq.10, we can identify that the loop \( L_2 \) and \( L_3 \) are valid and the loop \( L_2 \) is invalid.

As illustrated in Fig.7, when a polygonal contour offset some loops may result. In the case, some loops have to be removed while others are kept. From Fig.7, it can be determined intuitively that loops \( L_3 \) and \( L_4 \) should be removed and loops \( L_1, L_2 \) and \( L_3 \) should be kept.

Algorithm 2. Description of processing loops algorithm

Input: offset polygon list OutlineList contained a set of vertex of the polygon and a set of intersection point of offset contour.

Output: a set of separate loop.

Step1: Load Offset polygonal list and Intersection point list.

Step2: Search the Intersection point list, if the number of intersection point is zero then terminate the process; otherwise, go to Step3.

Step3: The offset polygon is traced along its positive direction starting from any one point (vertex \( P_i \) of Fig.10) until any intersection point is encountered (intersection point \( P_i \) in this example). At this instant, the travel direction is transferred to positive direction of another edge from intersection point (vertex \( P_k \) of Fig.10).

Step4: Repeated Step3 until reach the start point, this form a closed loop.

Step5: In light of formula (10) and formula (11), identify the loop invalid or valid. If the loop is invalid, then remove the loop; otherwise, store the loop as a separate polygonal contour.

Step6: Repeat Step3-Step5 until all process all loops.

V. ALGORITHM ANALYSIS AND EXAMPLE

A. Algorithm analysis

We now discuss the efficiency of the algorithm in this paper. The algorithm of laser path compensation consists of three phases:(1) cross-sectional contours offsetting; (2) intersection points detection; (3) invalid loops removal. For the first phase, vertexes of offset polygons are calculated one by one for polygonal contours. The time complexity of the phase is \( O(n) \), where \( n \) is the number of vertex of polygonal contours. The generated offset polygon avoids the sharp vertex with a simple method, so the proposed algorithm is much more efficient than that of literature[5].

To detect self-intersection points of offset polygon (second phase), the algorithm is proposed with time complexity \( O((n+k)\log m) \), where \( k \) is the number of intersection points and \( m \) is the number of monotone chains. The time complexity of Bentley-Ottmann’s sweep-line algorithm is \( O((n+k)\log n) \). Obviously, in common case, the number of monotone chain is much less than that of edges of polygon. The time complexity of proposed algorithm is the same as that of Sang[8]. But proposed algorithm does not need to split the polygonal contour and the time complexity of splitting polygonal contour is \( O(n+k\log n) \), so the algorithm is more efficient than that of Sang. For third phase, the time cost is \( O(n\log n) \). The algorithm presents a simple and uniform identifying method of the invalid loops.
B. Examples

The proposed algorithms have been implemented using MS Visual C++ programming languages.

Fig. 8 shows an offset contour of sectional contour. The offset contour has 10 intersection points and 6 invalid loops. After removal of invalid loops, 5 valid loops are obtained. Every valid loop treated as a separate contour, namely the laser path (see Fig. 8(b)).

Figure 8 Offset contour and removing invalid loops

As shown in Fig. 9(a), the STL model of a small cow toy is comprise of 5804 triangular facets. Its size of envelopment-box is 34x53x97mm. Fig. 9(b) shows polygonal contour and offset contour of No. 51 layer after the STL model is sliced using implemented slicing program. The offset polygon consists of 284 linear segments. There exit three self-intersection points in its forefoot of offset polygon, as shown in Fig. 9 (c). Fig. 9 (d) shows the offset polygon after invalid loops are removed.

Figure 9 Example of a cow toy

VI. CONCLUSION

In rapid prototyping technology, the loop problem is often appears and is one of most problems in laser path compensation. In this paper, we propose an effective algorithm generating the correct laser path for stereolithograph. The algorithm not only enhances manufacturing equality but also ensures later data processing such as contour filling. The contribution of this paper include:

(1) The calculating method of offset polygon is improved, which enhance the efficiency of the procedure. In application, the algorithm reveals a good robustness.

(2) A modified algorithm of intersection point detection and a algorithm of invalid loop removal are present.

(3) The ability of rapid prototyping system is enhanced to machine detailed portion of part.


