Filling Algorithm of Polygons and Embedded Polygons

Xiaozhuo Yang

1 School of Automation, Wuhan University of Technology, Wuhan, Hubei, 431700, China

*Corresponding author’s e-mail:290059@whut.edu.cn

Abstract. In this paper, by studying the laser marking trajectory model, for the two different marking modes of zigzag parallel and contour parallel, with or without contour nesting, different laser marking path generation algorithms are designed. And compare the time-consuming of the two algorithms, discuss and propose methods to optimize the algorithm to improve the efficiency of the algorithm to meet the needs of industrial applications. According to the requirements of the distance between the retracted boundary and the hatch line, the zigzag parallel trajectory generation algorithm and the contour parallel trajectory generation algorithm are designed, and the hatching curve is generated according to the given contour point set. Based on the algorithm above, in the face of contour nesting, the algorithm is improved, and the hatching curve is generated according to the given contour point set. For the zigzag parallel trajectory generation algorithm, considering the intersection of the scan line and the inner contour. For contour parallel trajectories, this paper uses a vertex-based offset algorithm, which takes a long time. This is because the vertex-based offset algorithm is more complicated to solve the problem of self-intersection. This paper proposes that for the zigzag parallel method, the angle of the workpiece can be adjusted to find the optimal angle, so that the marking path is the smallest. And for the contour parallel trajectory generation algorithm, this paper proposes an optimization too.

1. Introduction

Laser is an important invention in the 20th century. The hatch tool of laser marking machine can be used to hatch specified 2D-compound curve graph. The direction parallel hatch and the contour parallel hatch are two basic ways of hatching. The direction parallel hatch, also known as “zigzag” hatch, has paths being moved along line segments which are parallel to an initially selected reference direction. Based on this strategy a connected path is obtained by linking these parallel segments so that they are either all traversed from right to left (or left to right) or, alternately from left to right and from right to left. Whereas the contour parallel hatch uses offset segments base the boundary curves as smooth hatch path that similar to the boundary curve. Thus, the contour parallel hatch be generated in a spiral-like fashion along curves that are at constant distances from the curve boundary.

2. Model establishment and algorithm design

2.1. Offset Algorithm

The contour of the scanning area is determin-ed by a set of given coordinate points $S$. In order to solve the actual hatching curve area, an offset algorithm for the set of contour points $S$ is denned. As shown in figure 1.
The algorithm begins from the offsetting of an external cycle inward of offsetting the internal cycle outward. Without loss of generality, set the origin of the temporary coordinate system at the arbitrary vertices of the cycle $V_i$.

$L_i$ and $L_{i+1}$ are the vectors formed by $V_i$, $V_j$, and $V_{i+1}$.

Figure 1. Offset Generation

Let $L_i$ and $L_{i+1}$ be the equidistant line of $L_i$ and $L_{i+1}$. If the offset is R, then the following equations are derived, for the inward offsetting:

$$
\begin{align*}
\vec{I}_i &= x_i \hat{i} + y_i \hat{j} \\
\vec{I}_{i+1} &= x_{i+1} \hat{i} + y_{i+1} \hat{j}
\end{align*}
$$

Then:

$$
\begin{align*}
-x_i x + x_i y &= R \\
-x_{i+1} x + x_{i+1} y &= R
\end{align*}
$$

\[x = \frac{(x_{i+1} - x_i)R}{x_i y_i - y_i^2}
\]

2.2. zigzag parallel trajectory generate algorithm

According to the offset algorithm and the set of given coordinate points, the actual indented boundary contour is obtained. By scanning the coordinate points from left to right and top to bottom, each layer of the intersection of the indented contour and the horizontal hatch scan line can be obtained, and the points are divided into odd-numbered points by recording the order of each layer of intersection from left to right and even points.

Assume that the total number of layers is $n$ and the total number of intersection points is $m$, then the time complexity of Algorithm 1 is $O\left(\frac{n}{m-1} + \frac{m-1}{2n}\right)$.

2.3. contour parallel trajectory generate algorithm

When the current contour is known, based on the offset algorithm, an inward offset contour corresponding point set relative to the current contour can be generated to generate the next level of contour. As shown in figure 2, the contour generated by the offset algorithm will self-intersect.

The contour $L_0$ is the original contour, and the dashed contour line is the contour generated after the offset of $L_0$, which produces three self-intersecting loops $L_0$, $L_2$, $L_3$, and the point $P^*$ is the line segment. A self-intersection point of $P_i$, $P_{i+1}$ and $P_k$, $P_{k+1}$. To detect the self-intersection point of the generated offset contour, first determine whether the intersection point of each side of the contour...
exists. Take edge $P_i P_{i+1}$ and edge $P_k P_{k+1}$ as an example. If the intersection of the two edges exists, the coordinates of the intersection point $P^*$ can be calculated by the following formula:

$$P^* = P_i + tV_i = P_k + sV_2 \quad (0 \leq t \leq 1 \quad 0 \leq s \leq 1)$$

![Figure 2. Contour offset self-intersection](image)

![Figure 3. Contour nesting](image)

2.4. Model with contour nesting

For the case of contour nesting, define the number of layers of the contour: if the contour $a$ is surrounded by $n$ layers of contour, then the number of layers of the contour $a$ is $n+1$. As the figure 3 shows, in the case of nested contours, the blue contours $a$, $d$, $e$ are the outer contours, and the number of layers is 1, 3, 3, which are all odd numbers, and the red contours $b$, $c$, $g$, $f$ are the inner contour, and the number of layers is 2, 2, 4, 4, all of which are even numbers.

The odd-numbered layer contour is the outer contour of the scanning area, and the even-numbered layer contour is the inner contour of the scanning area. Define the positive traversal direction of the outer contour as clockwise, and the positive traversal direction of the inner contour as counterclockwise.

3. Model solution

3.1. Solving sawtooth parallel model

While internal contraction of boundary distance $R_1 = 1\text{mm}$, hatch line spacing $R_2 = 1\text{mm}$. The zigzag parallel trajectory generation algorithm is applied to realize the generation of zigzag parallel trajectory, and the effect is shown in the figure 4.

While internal contraction of boundary distance $R_1 = 0.1\text{mm}$, hatch line spacing $R_2 = 0.1\text{mm}$, realize the generation of zigzag parallel Trajectory. The number of generated zigzag parallel lines, the average time of incubation program, and the number of zigzag parallel tracks are shown in the table 1.

When solving sawtooth parallel model with contour nesting, the effect is shown in the figure 5.

The number of zigzag parallel lines generated, the average time of incubation program, and the number of zigzag parallel tracks are shown in the table 2.
Figure 4 zigzag trajectory $R_1=1\text{mm}$, hatch line spacing $R_2=1\text{mm}$

Figure 5 mutually-nested zigzag trajectory $R_1=1\text{mm}$, hatch line spacing $R=1\text{mm}$

Table 1. Sawtooth parallel trajectory

| $R_1$ | $R_2$ | Line width | Number of zigzag parallel lines) | Number of zigzag parallel tracks | Total length of zigzag parallel lines | Average runtime |
|-------|-------|------------|----------------------------------|----------------------------------|---------------------------------------|-----------------|
| 1\text{mm} | 1\text{mm} | 0.1\text{mm} | 85                               | 5                                | 888.975\text{mm}                     | 0.75s           |
| 0.1\text{mm} | 0.1\text{mm} | 0.01\text{mm} | 899                              | 5                                | 10010.119\text{mm}                   | 13.45s          |

Table 2. mutually-nested zigzag trajectory

| $R_1$ | $R_2$ | Line width | Number of zigzag parallel lines) | Number of zigzag parallel tracks | Total length of zigzag parallel lines | Average runtime |
|-------|-------|------------|----------------------------------|----------------------------------|---------------------------------------|-----------------|
| 1\text{mm} | 1\text{mm} | 0.1\text{mm} | 103                              | 8                                | 772.103\text{mm}                     | 1.03s           |
| 0.1\text{mm} | 0.1\text{mm} | 0.01\text{mm} | 1065                             | 6                                | 9014.264\text{mm}                    | 31.65s          |

3.2. Contour parallel mode solution

While internal contraction of boundary distance $R_1 = 1\text{mm}$, hatch line spacing $R_2 = 1\text{mm}$. The contour parallel trajectory generation algorithm is applied to realize the generation of contour parallel trajectory, and the effect is shown in figure 6.

The number of contour coils generated, the average time of incubation program, and other data are shown in the table 3.

When solving contour parallel model with contour nesting, the effect is shown in the figure 7.

The number of parallel lines generated, the average time of incubation program, and the number of parallel tracks are shown in the table 4.

Table 3. mutually-nested zigzag trajectory

| $R_1$ | $R_2$ | Line width | Number of contour coils | Total length of contour | Average runtime |
|-------|-------|------------|-------------------------|-------------------------|-----------------|
| 1\text{mm} | 1\text{mm} | 0.1\text{mm} | 8                       | 896.821\text{mm}       | 1.03s           |
| 0.1\text{mm} | 0.1\text{mm} | 0.01\text{mm} | 6                       | 10017.094\text{mm}     | 31.65s          |

Table 4. mutually-nested contour trajectory

| $R_1$ | $R_2$ | Line width | Number of contour coils | Total length of contour | Average runtime |
|-------|-------|------------|-------------------------|-------------------------|-----------------|
| 1\text{mm} | 1\text{mm} | 0.1\text{mm} | 5                       | 1068.838\text{mm}      | 10.25s          |
| 0.1\text{mm} | 0.1\text{mm} | 0.01\text{mm} | 59                      | 9060.123\text{mm}     | 131.20s         |
4. Optimization of contour parallel trajectory generation algorithm

At present, the contour offset algorithm is mainly based on the skeleton line extraction algorithm and the isometric line algorithm. Voronoi diagram is generally used to extract the skeleton line and then the bias trajectory is obtained through the skeleton line. This algorithm has a high time complexity in the multi-connected region. Self-intersection, and cross-intersection will occur in the isometric line algorithm, removing the complexity.

The isometric line algorithm mainly includes two aspects. One is the isometric bias of contour line, which is mainly divided into vertex bias and line segment global bias. Vertex bias algorithm has high efficiency but low robustness, and there are cases of local sharp angles and invalid polygons after the removal of self-intersection. However, every point of the line segment overall bias should be offset twice, which makes the whole bias process less efficient but more robust.

Based on vertex offset algorithm computational complexity lies mainly in contour intersection and the determination of effective ring offset algorithm based on vertex need to find the intersection point, determine the intersection ring, whether selfing ring effectively, at the same time there will be offset vector problem such as reverse phase, local Angle, while the offset algorithm-based online segment can effectively avoid these problems.

To improve the efficiency of the algorithm, to meet the needs of industrial production, can use two kinds of synthesis methods, hybrid offset method based on vertex line design, when not in use vertex offset algorithm computation, when the intersection appeared, using line offset algorithm calculation, which can effectively avoid selfing calculation and judgment of the loop, significantly improve the efficiency of the algorithm, further meet the needs of industrial production.

5. Conclusion

This paper, for the two different marking modes of zigzag parallel and contour parallel, with or without contour nesting, compares the time-consuming of the two laser marking path generation algorithms.

And this paper proposes that for the zigzag parallel method, to find the best angle of the workpiece can improve the efficiency of the algorithm significantly. For the contour parallel method, hybrid offset method based on vertex line design, when not in use vertex offset algorithm computation, when the intersection appeared, using line offset algorithm calculation.

References
[1] Jin, Y., He, Y., Xue, G., Fu, J. (2015). A parallel-based path generation method for fused deposition modeling. The International Journal of Advanced Manufacturing Technology, 77, 927-937.
[2] Jin, Y., He, Y., Du, J. (2017). A novel path planning methodology for extrusion-based additive manufacturing of thin-walled parts. International Journal of Computer Integrated Manufacturing, 30, 1301-1315.

[3] Jiang, J., Ma, Y. (2020). Path Planning Strategies to Optimize Accuracy, Quality, Build Time and Material Use in Additive Manufacturing: A Review. Micromachines, 11.

[4] Ding, D., Pan, Z., Cuiuri, D., Li, H. (2014). A tool-path generation strategy for wire and arc additive manufacturing. The International Journal of Advanced Manufacturing Technology, 73, 173-183.

[5] Ding, D., Pan, Z., Cuiuri, D., Li, H. (2015). A practical path planning methodology for wire and arc additive manufacturing of thin-walled structures. Robotics and Computer-integrated Manufacturing, 34, 8-19.

[6] Kim, H. (2010). Optimum tool path generation for 2.5D direction-parallel milling with incomplete mesh model. Journal of Mechanical Science and Technology, 24, 1019-1027.

[7] 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, 657-670.

[8] Held, M. (2005). A geometry-based investigation of the tool path generation for zigzag pocket machining. The Visual Computer, 7, 296-308.

[9] Lin, S., Xia, L., Ma, G., Zhou, S., Xie, Y.M. (2019). A maze-like path generation scheme for fused deposition modeling. The International Journal of Advanced Manufacturing Technology, 104, 1509-1519.