Team Number:	2020180010075

Problem Chosen:	A

2020 APMCM summary sheet

This question requires the use of lasers to fill hatches in closed curve polygons and multiple mutually-nested contour objects. For the graphic hatching process, there are two methods: Zigzag and Contour-parallel. It is an optimization problem of complete coverage path planning (CCPP) in barrier areas. The full coverage path planning of the barrier-containing area is divided into two issues: regional segmentation and partition path planning.

For question 1 and 2, there are 2 main steps. **Step1: Pattern preprocessing**. The data in attachments 1 and 2 need to be converted into scatter plots, and then the scatter plots are fitted to a closed curve. Then the closed curve is offset inward, and finally the offset patterns are environment modeled by the Grid Method. **Step2: Path filling**. For a single-layer contour pattern, if the hatch strategy is zigzag parallel, the pattern should be partitioned first, and then path planning is performed in the sub-region to find the shortest path. The Contour parallel hatch method directly performs path planning. For multi-layer patterns, we need first distinguish the inner and outer contours to obtain the area that needs to be marked, and then plan the path. In the path planning part, A* algorithm was used to find the optimal paths. Results of question 1 and 2 are as follows:

	Hatch ways	Boundary distance	Total length(mm)	Horizontal lines/Circles	Average elapsed time(ms)	Elapsed time ratio
Single-layer	Zigzag	1mm	893.058	85	324.6	30.49
		0.1mm	10009.413	901	9900.9	
	Contour	1mm	895.954	14	819.6	21.98
		0.1mm	10017.469	132	18015.9	
Multi-layer	Zigzag	1mm	742.842	89	1503.5	7.19
		0.1mm	9049.608	1056	10816.5	
	Contour	1mm	766.781	18	2712.3	8.48
		0.1mm	9056.616	198	23007.4	

In order to improve the efficiency and performance of the model, we consider the optimization of partition join order and the improvement of planning algorithm. The former is to optimize multiple subregions as A multi-travel agent problem, while the latter is to integrate the advantages of traditional artificial potential field method and A* algorithm.

Key words: Outline offset; A* algorithm; Path planning; Rasterization; Regional segmentation

Contents

1 Introduction

1.1 Background

Laser marking is the industrial application of laser technology at the end of the 20th century. It is the perfect combination of laser technology and computer technology. Before laser marking technology has not been applied, laser is generally used for welding, cutting, heat treatment and other industrial processing [1]. With the development of computer technology and embedded technology, laser has gradually been used for industrial marking [2]. The working principle of laser marking is to use high energy and high-density laser beam to act on the marked object surface. As the laser beam moves, the marked path is formed on the object surface, so as to obtain the visible pattern.

At the same time, because the laser marking does not contact the processing material, it does not produce any mechanical deformation on the surface of the workpiece, so that its application range has been greatly expanded. Most importantly, with the development of computer and electronic technology, laser marking combining laser technology and computer technology can modify the content of the tag in the computer, which is favored by users.

Zigzag has been the most widely adopted fill pattern due to its simplicity. However, a zigzag fill consists of many sharp turns, a problem that is amplified when printing shapes with complex boundaries or hollow structures.[3] A contour-parallel toolpath, formed by iso-contours of the Euclidean distance transform, provides a remedy, but it leads to high contour plurality since theism-contours are disconnected from each other. A spiral fill pat-tern, for simple shapes such as a square, is continuous.

In recent years, with the improvement of laser performance and reliability, coupled with the rapid development of computer technology and the progress of optical devices, laser marking technology has developed very rapidly.[4]

1.2 Restatement of the problem

This question requires the use of lasers to fill hatches in closed curve polygons and multiple mutually-nested contour objects. For the graphic hatching process, there are two methods: Zigzag and Contour-parallel. First, offset the existing boundary contour inward or outward according to the edge distance, and then perform zigzag parallel or contour parallel shading. The shadow curve should be uniform, regular, and basically parallel. Neither omission of filling in a certain area nor repeated filling of the area is allowed. The laser-marked shadow should generate the shadow contour online in real time. In order to meet the requirements of high-efficiency laser marking, the shadow curve should be kept parallel to the boundary line of the graph, distributed evenly to the maximum extent, and can be automatically and quickly generated.

After studying the characteristics of using zigzag parallel and contour parallel shadows in the pictures provided in the title, establish the mathematical model of the shadow, design the algorithm, discuss the efficiency of the algorithm, and answer the following questions:

Implement the zigzag parallel and contour parallel hatch of the single-layer contour pattern in Attachment 1 and multi-layer contour pattern of the Attachment 2. And calculate the number of horizontal lines of the zigzag parallel shadow and the number of circles of the parallel shadow of the contour. Calculate the average running time according to the multiple runs of the incubation program, and calculate the running time ratio of the program under the conditions of parameter groups (2) and (1).

In order to meet the efficiency requirements of actual industrial applications, please provide optimization strategies or directions for the performance and efficiency of the hatching algorithm.

2 Analysis of the problem

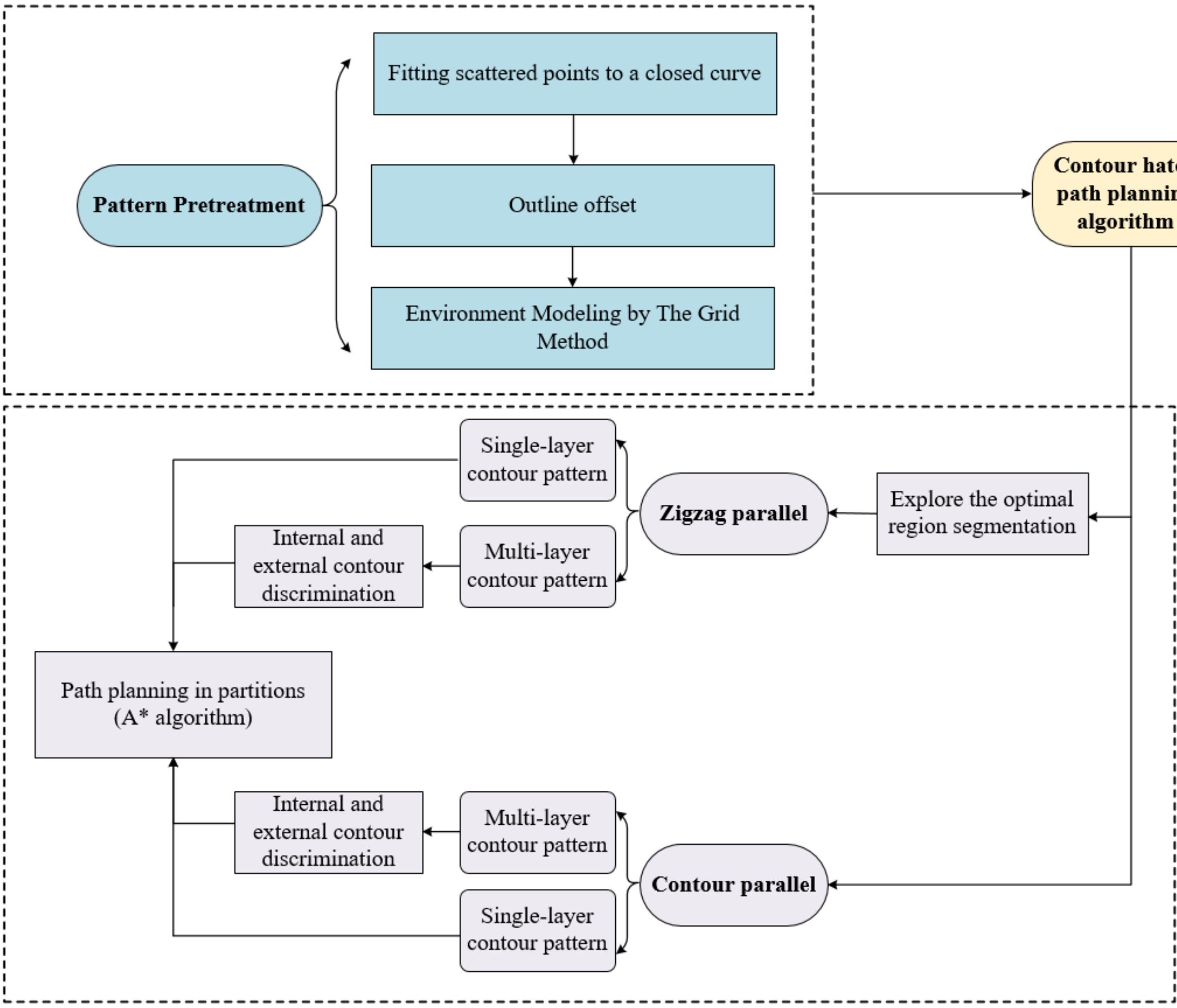


Fig.1 Flow chart of the full text

This question requires the use of lasers to fill hatches in closed curve polygons and multiple mutually-nested contour objects. For the graphic hatching process, there are two methods: Zigzag and Contour-parallel.

Step1: Pattern preprocessing. The data in attachments 1 and 2 need to be converted into scatter plots, and then the scatter plots are fitted to a closed curve. Then the closed curve is offset inward, and finally the offset patterns are environment modeled by the Grid Method.

Step2: Path filling. For a single-layer contour pattern, if the hatch strategy is zigzag parallel, the pattern should be partitioned first, and then path planning is performed in the sub-region to find the shortest path. The Contour parallel hatch method directly performs path planning. For multi-layer patterns, we need first distinguish the inner and outer contours to obtain the area that needs to be marked, and then plan the path.

The task of laser marking full coverage can be abstracted as marking the pattern with obstacles. It is hoped that the laser can complete the full coverage task with the minimum cost, such as the shortest distance, the shortest time, the lowest energy consumption, or the minimum of the sum of these costs. Obviously, this is an optimization problem of complete coverage path planning (CCPP) in barrier areas. The full coverage path planning of the barrier-containing area is divided into two issues: regional segmentation and partition path planning.

Question 1 and question 2 are to achieve the goal of as few laser extinguishing times as possible and running time as short as possible. The respective optimization of the above two issues and the overall optimization of the entire barrier area full coverage task are considered. This article follows this principle to study the problem.

3 Assumptions

To conveniently solve the laser marking problem, we make the following assumptions:

1. Assuming that the contour parallel mark is marked counterclockwise.
2. Assuming that the initial mark direction of the zigzag parallel mark is to the right.
3. Assuming the highest point in the sub-area is the starting point of the search after partitioning.
4. Assuming that when the inner and outer contours conflict, take the curved path first, and partition when the conflict cannot be resolved.
5. Assuming that the laser extinguishing and starting time is ignored when calculating the time.
6. Assuming that when calculating the zigzag parallel mark, the parallel of different partitions are counted as different parallels, which need to be calculated separately.
7. Assuming that when calculating the number of contour circles, the number of circles in different partitions counts as different circles, and the circles that have been given in the article do not participate in the calculation.

4 Symbol description

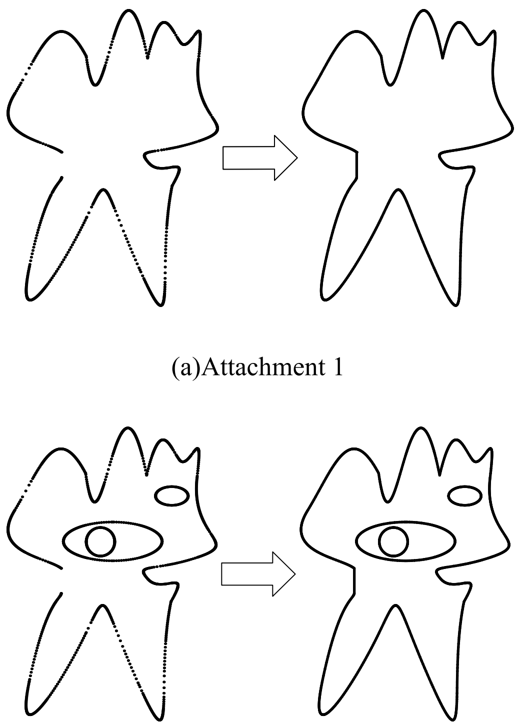
This chapter introduces the symbols and their descriptions mentioned in the path search modeling process of the two marking strategies Zigzag parallel and Contour parallel.

	Symbol	Description
Shared symbols	Ω	The fitted pattern of the scatter plot obtained from the attached data
	Ω_p	New contour obtained after offsetting contour Ω
	d	Boundary distance
Zigzag parallel	l_i	Horizontal line
	C_{2i-1}	Nodes that l_i intersects the left offset contour Ω_p
	C_{2i}	Nodes that l_i intersects the right offset contour Ω_p
Contour parallel	A_i	The grid set where $d_i \leqslant d$,the left grid set is A_1, and the right grid set is A_2
	P_i	The highest point of the innermost grid
	P_j	The lowest point of the innermost grid

5 Pattern preprocessing

5.1 Fitting scattered points to a closed curve

The data in Attachment 1 and Attachment 2 are 1,195 coordinate points and 1,681 coordinate points. After these points are marked in the coordinate system, they become two scatter plots. Because the hatching entities must be closed curvilinear polygons, connect the dots one by one to form a closed curve. The scatter plot drawn from the data in Attachment 1 and the fitted closed pattern are shown in Figure 3(a). Similarly, the scatter plot drawn from the data in Attachment 2 and the closed pattern after fitting are shown in Figure 3(b).



(a)Attachment 1

(b)Attachment 2

Fig.3 The scatter plots drawn from the data and the fitted closed patterns

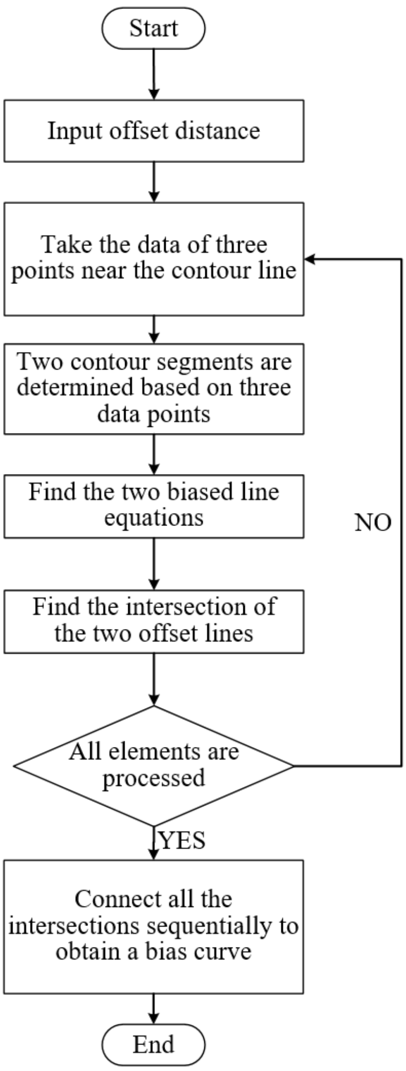


Fig.4 Outline equidistant offset flow chart

5.2 Outline offset

Due to the requirement of boundary distance in the question, after fitting the initial outline, we carried out an operation to offset the outer outline inward again, so as to ensure the boundary distance scanning conditions of 1mm and 0.1mm. The outer outline bias here is actually an equidistant bias algorithm. The essence of the algorithm is to divide the irregular outer outline into outline segments based on the idea of using limit, and require the parallel lines of each outline segment to be offset internally to fix the same distance, and then obtain the new offset outline by intersecting the offset parallel lines. This method can effectively guarantee the geometric features of the external outline of the model.

Outline offset is actually a parallel offset of the regional outline line along different directions. The process of offset is as follows:

It is assumed that the outline line can be divided into countless intersecting line segments. When the included Angle of the line segment gradually becomes peaceful, the outline line is considered to be peaceful. In the outline bias process, we can take any two-line segments and observe the bias process. As shown in the figure 5, we offset Q_1Q_2 and Q_1Q_3 by distance d respectively to obtain two parallel new line segments, and set the coordinate as $Q_1(x_1,y_1), Q_2(x_2,y_2), Q_1'(x_1',y_1'), Q_2'(x_2',y_2')$.

According to the known parallel conditions, it can be obtained as follows:

$$\begin{cases} \vec{D}_1\vec{D}_2 \cdot \vec{D}_1'\vec{D}_2' = 0 \\ \left|\vec{D}_1\vec{D}_1'\right| = d \end{cases} \tag{1}$$

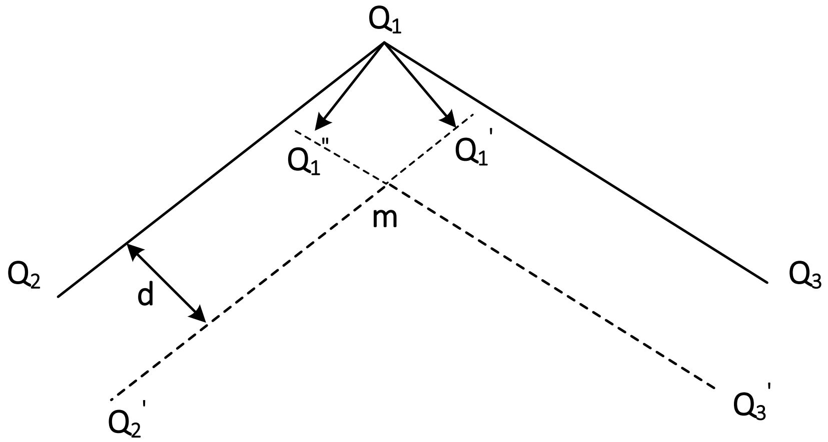


Fig.5 Outline offset diagram

Put the coordinates into the formula to get:

$$\begin{cases} (x_2 - x_1)(x_2 - x_2) + (y_2 - y_1)(y_2' - y_2) = 0 \\ (x_2' - x_2)^2 + (y_2 - y_2)^2 = d^2 \end{cases} \tag{2}$$

We can figure out the coordinates $Q_2'(x_2',y_2')$ according to the formula, and similarly, we figure out the coordinates $Q_1'(x_1',y_1')$, and we get the endpoint information of the offset segment $Q_1'Q_2'$. In the same way, the coordinate $Q_1''(x_1'',y_1'')$

and $Q_3'(x_3',y_3')$ are obtained, that is, we get the endpoint information of the offset segment $Q_1''Q_3'$.

According to the above formula to find the endpoints of both ends of the line segment, the next step is to find the coordinates of the intersection point m, in fact, to find the intersection point of the offset line.

After outline bias is set, the new equation of the line $Q_1'Q_2'$ is:

$$y_{L_1} = kx_{L_1} + b \tag{3}$$

where:

$$k = \frac{\Delta y'}{\Delta x'} = \frac{y_2' - y_1'}{x_2' - x_1'} \, (x_2' \neq x_1') \tag{4}$$

Substituting one point into the equation of the line can be obtained as follows:

$$b = \frac{x_2' y_1' - x_1' y_2'}{x_2' - x_1'} \tag{5}$$

The line equation of offset outline segment $Q_1'Q_2'$ can be obtained as follows:

$$y_{L_1} = \frac{y_2' - y_1'}{x_2' - x_1'} x_{L_1} + \frac{x_2' y_1' - x_1' y_2'}{x_2' - x_1'} \tag{6}$$

And once again, we can get $Q_1''Q_3'$:

$$y_{L_2} = \frac{y_2' - y_3'}{x_2' - x_3'} x_{L_2} + \frac{x_2' y_3' - x_3' y_2''}{x_2'' - x_3'} \tag{7}$$

According to the equation of two new outline offset lines, we find the intersection point m coordinates. Similarly, for a smooth outline line, we can do multiple segmentation, and find the intersection point of the intersecting offset line through this method, and connect the intersection points obtained in accordance with the order to obtain the outline offset curve.

According to the parameters provided in Attachment 1, 1mm and 0.1mm profile bias were carried out for the single-layer profile to obtain the results shown in the figure 6, which laid a foundation for further shadow scanning.

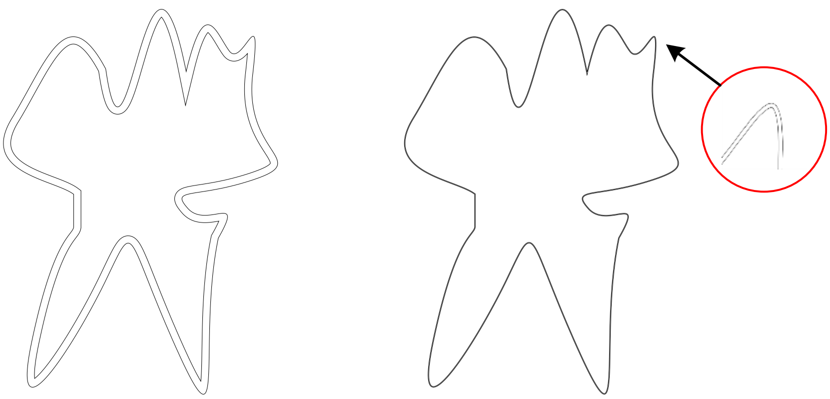


Fig.6 Internal contraction of boundary distance 1mm(right),0.1mm(left)

5.3 Environment Modeling by The Grid Method

Environmental representation problem refers to the representation of obstacles in the environment and free space. Environment modeling is an effective description of the moving target activity space and the representation of obstacles and contours. For example, grid method environment modeling is shown in Figure 7. Only reasonable environmental representation can help reduce the search volume in planning and reduce the cost of space and time. Different planning approaches are modeled based on a variety of different environments.

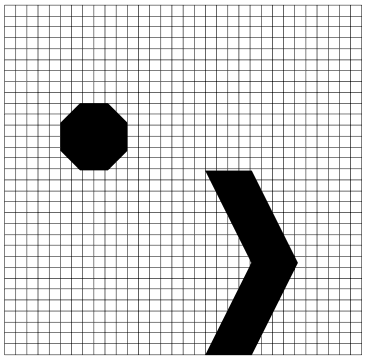


Fig 7 Grid method environment modeling diagram

In this paper, Lasers have good directivity and can concentrate into small spots of light. The small spot formed by the laser is approximated as a small grid, and the appropriate grid size is further selected to carry out the environmental modeling using the grid method.

Specifically, the grid map model is to use a certain mathematical model to represent the working area of the laser. Map rasterization is the division of the area where the laser performs the hatch scanning task into a series of raster square structures of the same size. The position of raster coordinate system can be accurately located, which is convenient for environment modeling and algorithm planning.

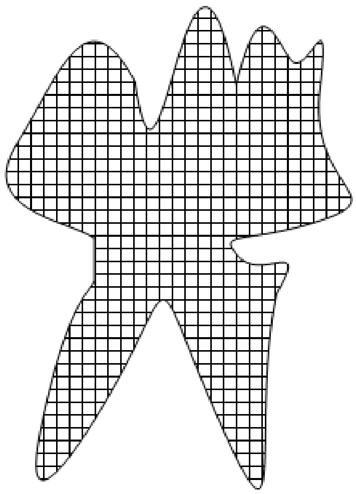


Fig.8 Grid method environment modeling for single-layer contour pattern

We rasterized the single-layer contour pattern as shown in Figure 8. We assume that the minimum side length of the grid is **0.001** mm. In order to objectively represent the environment of the grid method, the following definitions are made:

- **Definition 1**: $A\{(x,y) \mid (x,y)\ \textit{represent all the point coordinates}\}$
- **Definition 2**: Before the laser starts hatch scanning, we define the values of the grids as equation follows, and we can obtain the area where can reach $S_{\phi}=\{x,y \mid ocup(x,y)=1\} \subseteq S$, and we can obtain the area where can't reach $S_{\varDelta}=\{x,y \mid ocup(x,y)=0\} \subseteq S$.

$$ocup(x,y)=\begin{cases}1, & can\ reach\\ 0, & can't\ reach\end{cases} \tag{8}$$

- **Definition 3**: After the laser starts hatch scanning, we define the values of the grids from $ocup(x,y)=1$ to $ocup(x,y)=2$.

6 Introduction of Zigzag parallel hatch and Contour parallel hatch

On the premise of no missing traversing sub-region, the criteria to measure the advantage and disadvantage of the inner walking strategy are time and distance. The main factor affecting the time is the number of turns, the more turns, the more time. The main factor affecting the distance is the walking strategy, which is the main way of shadow scanning path planning.

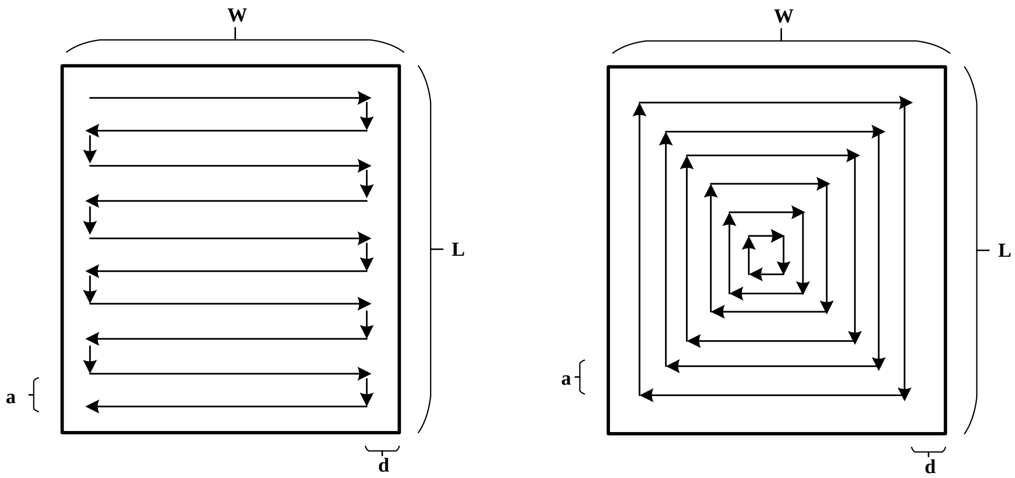


Fig. 9 the directional parallel hatch (right) and the contour parallel hatch (left)

In this paper, the hatch walking strategy selection is mainly aimed at reducing the number of turns and the walking distance of laser scanning path, and the two hatch scanning walking strategies adopted in the problem are the traditional path planning algorithms in the filling problem: the directional parallel hatch strategy and the contour parallel hatch strategy.

In order to understand and compare these two traditional path planning strategies in Figure 9, we assume that a rectangular region has a width of W, a length of L, and a laser scan line interval of d. And We calculate the total path and turn number of these two scanning hatch walking strategies.

6.1 Zigzag parallel hatch

The direction parallel hatch, also known as "zigzag" parallel 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. And Figure 9 illustrates the zigzag hatch process. The direction parallel hatch is one of the simplest path planning algorithms, and its technology is relatively mature. Generally, scanning along the X-axis or Y-axis is the most commonly used method. For the right picture in Figure 9, we can see the total path S_{zzp} and the turn number T_{zzp} as follow:

$$S_{zzp} = \left[int\left(\frac{L-2d}{a} \right) + 1 \right] W + L - 2\mathrm{d} \tag{9}$$

$$T_{zzp} = 2int\left(\frac{L-2d}{a} \right) \tag{10}$$

And we usually adopt the directional parallel hatch according to the following advantages:

- ✧ Simple implementation and reliable operation.
- ✧ Relative to parallel straight-line scanning, its empty stroke is reduced.

6.2 Contour parallel hatch

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. And Figure 11 illustrates the contour parallel hatch process.

Contour bias method is based on the external and internal contours of the cross-section contour, and the internal bias towards the region with fixed distance is generated to generate the laser scanning path, as the left example shown in Figure 9. Because the whole area is a closed polygon, and each shadow scan path is within the contour boundary, there is no crossing of the contour. However, when the contour of the whole region is complex or the inner contour and the outer contour are close to each other, it will inevitably result in interleaving, which greatly increases the complexity of the computation of the offset contour. For the left example in Figure 9, we can see the total path S_{cph} and the turn number T_{zzp} as follow:

$$S_{cph} = 2\left[ML - 2d\left(\frac{(1+M)}{2} \right) M \right] + 2\left[MW - 2d\left(\frac{(1+M)}{2} \right) M \right] \tag{11}$$

$$M = Min\left\{ int\left(\frac{W}{2d} \right),\ int\left(\frac{L}{2d} \right) \right\} \tag{12}$$

Simplify the equation, we can obtain:

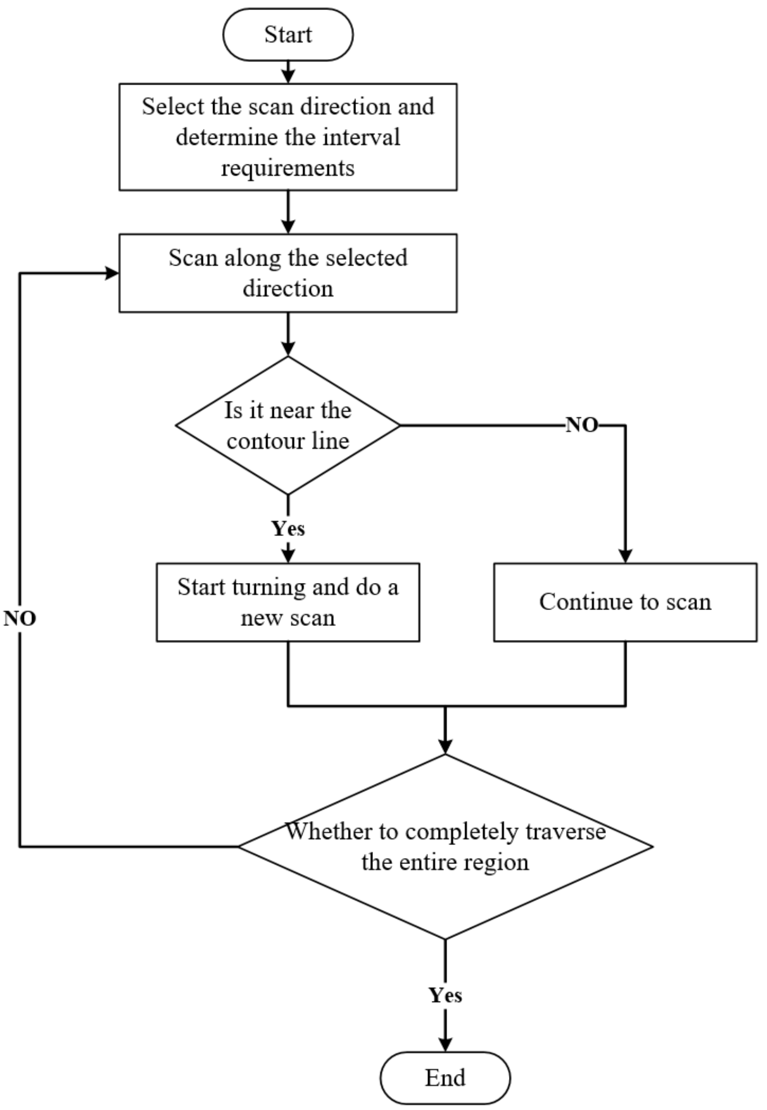


Fig.10 Zigzag parallel hatch process flow chart

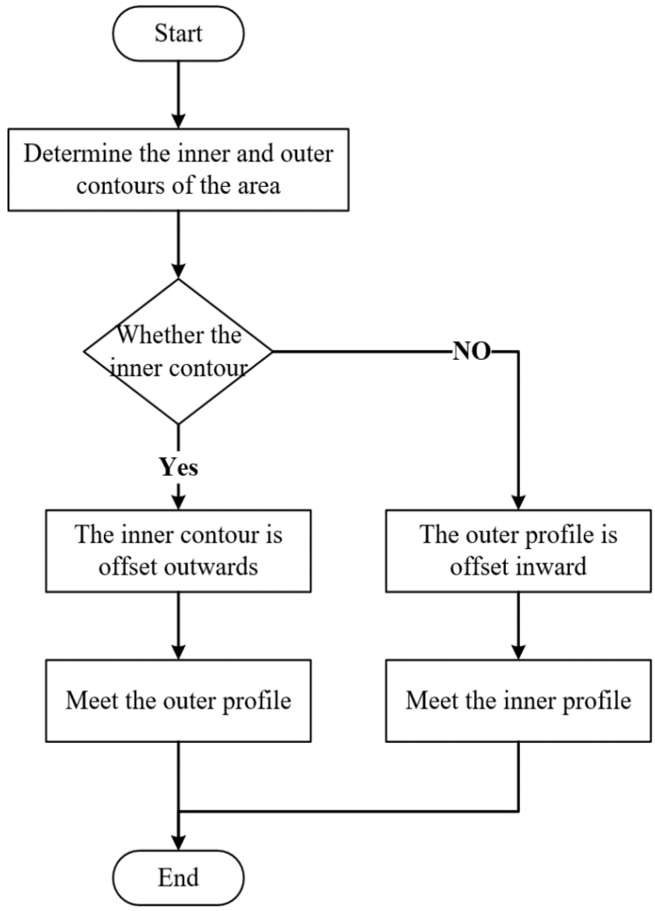


Fig.11 Contour parallel hatch process flow chart

$$S_{cph} = 2M[L + W - 2(1 + M)d] \tag{13}$$

$$M = Min\left\{int\left(\frac{W}{2d}\right),\ int\left(\frac{L}{2d}\right)\right\} \tag{14}$$

$$T_{cph} = 4Min\left\{int\left(\frac{W}{2d}\right), int\left(\frac{L}{2d}\right)\right\} \tag{15}$$

And we usually adopt the contour parallel hatch strategy according to the following advantages:

- ✧ Relatively smooth boundary.
- ✧ Compared with z-scan algorithm, burrs are reduced.
- ✧ The bias profile is suitable for continuous simple shapes.

7 Model building and solution of question 1

The task of laser marking full coverage can be abstracted as marking the pattern with obstacles. It is hoped that the laser can complete the full coverage task with the minimum cost, such as the shortest distance, the shortest time, the lowest energy consumption, or the minimum of the sum of these costs. Obviously, this is an optimization problem of complete coverage path planning (CCPP) in barrier areas. The full coverage path planning of the barrier-containing area is divided into three issues: regional segmentation, partition path planning, and partition hatch sequence.

Question 1 and question 2 are to achieve the goal of as few laser extinguishing times as possible and running time as short as possible. The respective optimization of the above three issues and the overall optimization of the entire barrier area full coverage task are considered. This article follows this principle to study the problem.

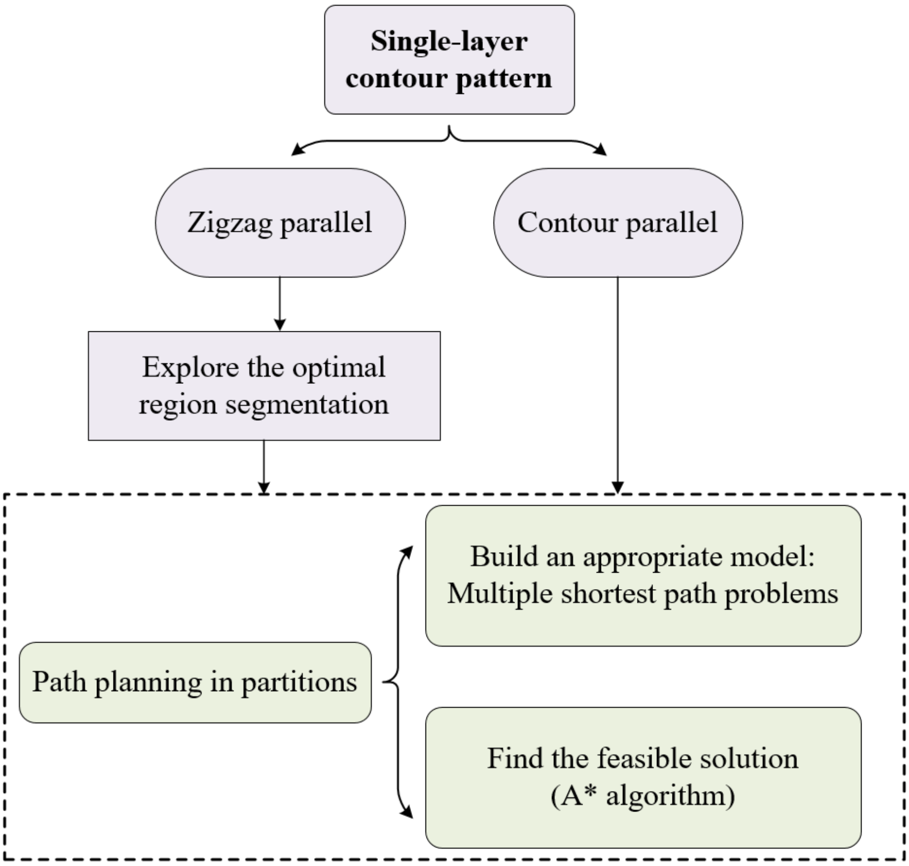


Fig.12 Flow chart of the solution of question 1

7.1 Zigzag parallel hatch of the single-layer contour pattern

7.1.1 Regional Convex segmentation

Since the pattern to be marked is a concave polygon, the laser cannot complete the marking of it at one time. It needs to be extinguished multiple times and the marking start point is reselected to complete the marking of all regions. Due to the high energy of the laser, frequent extinguishing will cause high energy loss. Therefore, when decomposing the pattern, the number of laser extinguishing should be as few as possible, that is, the number of partitions should be as few as possible. Partition marking is to decompose the pattern to be marked first, and then fill different regions separately. Therefore, the convex segmentation of the pattern to be marked is required first.

The region segmentation method is to divide the whole space region to form several partitions with simple shapes and no obstacles and overlaps. The coverage of each sub-area becomes a simple reciprocating motion, the work order of each sub-area is reasonably allocated, and the connecting route between partitions is optimized to complete the planning of the whole path.

- **Trapezoidal method**

Latombe et al[4] put forward trapezoid segmentation method that assumed a vertical "cutting line" from left to right scan through the whole area, cutting line and area there are a lot of tangent polygon obstacle, intersection and intercross event occurs, the obstacle of the area of the part, in turn, be divided into multiple partition, each region is trapezoidal[5]. As shown in figure 13(a), in a quadrilateral region a polygon obstacle, trapezoid segmentation method is used to analyze the regional segmentation, as shown in figure 13(b), were divided into 5 partition. Similarly, Figure 13(c) is divided into six partition, as shown in figure 13(d).

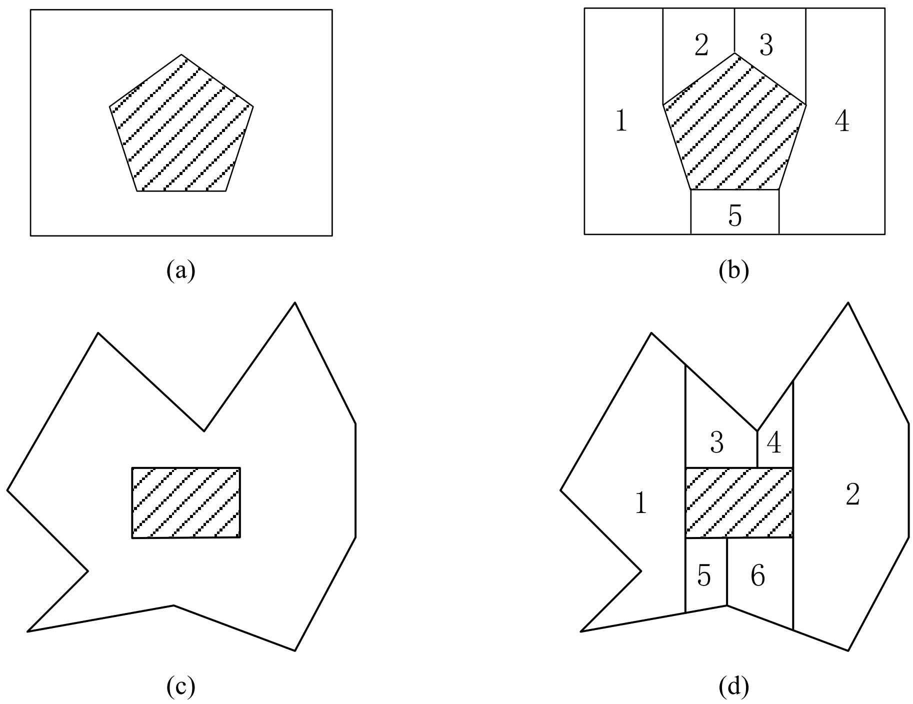
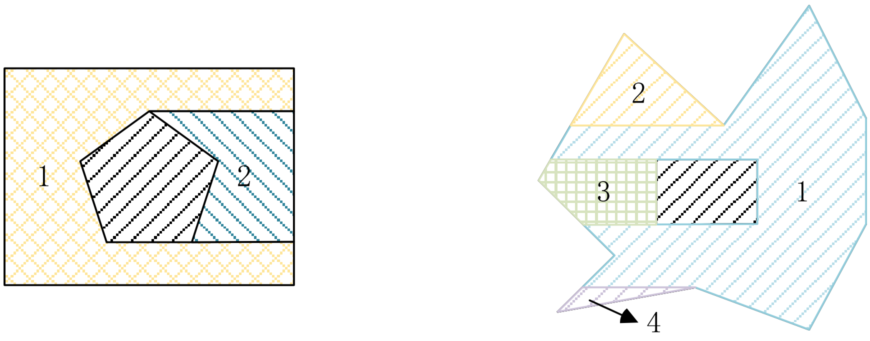


Fig.13 Schematic diagram of region segmentation by trapezoidal segmentation method

- **Maximum depth regional segmentation**

The maximum depth regional segmentation method is to select the maximum depth of the pattern as a region, and then select the second depth as a region, until all the areas of the pattern are divided.

As shown in figure 13(a), in a quadrilateral region a polygon obstacle, maximum depth regional segmentation method is used to analyze the regional segmentation, as shown in figure 14(a), were divided into 2 partition. Similarly, Figure 13(c) is divided into four partition, as shown in figure 14(b).



(a) (b)

Fig. 14 Schematic diagram of region segmentation by maximum depth regional segmentation method

Although the trapezoidal segmentation method is more convenient and faster, sometimes too many regions are decomposed to require more connecting routes, which not only makes the route planning more complicated, but also increases the length of the total covering routes.

As for the regional segmentation of figure 13(a) and figure 13(c), the number of partitions obtained by the maximum depth regional segmentation method and the trapezoidal segmentation method is compared as shown in Table 1 below.

Table 1 The comparison of the number of partitions obtained by the maximum depth region segmentation method and the trapezoidal decomposition method

	Number of partitions	
	Trapezoidal segmentation	Maximum depth regional segmentation
Fig2(a)	5	2
Fig2(c)	6	4

As shown in Table 1, it can be concluded that the maximum depth regional segmentation method has a better segmentation effect, with fewer partition obtained and fewer laser extinction times.

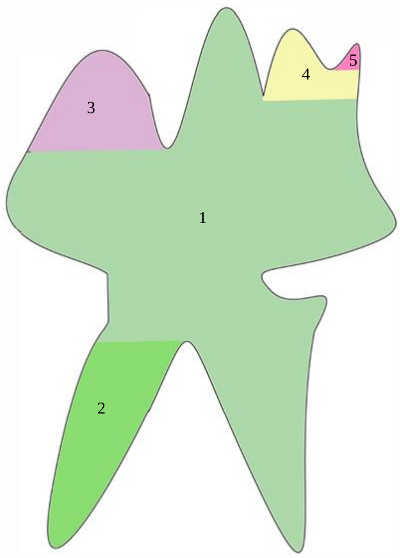


Fig.15 Partition map of single-layer pattern drawn according to Attachment 1

In conclusion, the regional segmentation method for Zigzag parallel hatch is the maximum depth segmentation method, and the resulting partition is shown in Figure 15 below. There are 5 partitions in the single-layer pattern drawn according to Attachment 1.

7.1.2 Path planning model in partitions

7.1.2.1 Model building

Path search is to use search technology to search and judge in a certain space, and then make a decision to find a path. Path search is to detect the surrounding environment through its own sensor in the actual environment, and to find a path through the search technology in the established environment model. Complete traversal path planning can be divided into multiple point-to-point path planning, and multiple point-to-point path planning constitute complete traversal path planning in accordance with a certain order.

The path search method of Zigzag Parallel hatch is explained as follows:

The path search of this problem can be subdivided into multiple shortest path problems.

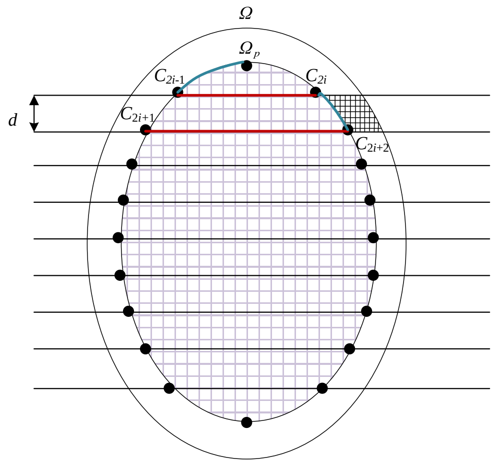


Fig.16 Search strategy of zigzag parallel hatch

Step 1: Take the highest point of the offset contour Ω_p obtained in 4.2 as the starting point of the path.

Step 2: According to the spacing width d, we draw the horizontal line l_i, and l_i intersects the offset contour Ω_p. They have two points of intersection, where the left one is C_{2i-1}, and the right one is C_{2i}. These intersections are the starting points for multiple shortest path problems.

Step 3: Let's start with C_{2i-1} and end with C_{2i}, search the inner grids of the offset contour Ω_p to find the shortest path between two points.

Step 4: Let's start with C_{2i} and end with C_{2i+2}, search the inner grids between the offset contour Ω_p and the contour Ω to find the shortest path between two points.

Step 5: Restart step 3 and 4 until you reach the lowest point in the region.

Step 6: End

7.1.2.2 Optimization algorithm

There are three main types of algorithms to solve the shortest path problem: The first is based on graph theory, such as Dijkstra and its improved algorithm, Floyd algorithm, etc.

The second is algorithms based on traditional artificial intelligence theory, such as A* algorithm and its improved algorithm, depth first search algorithm (DFS), breadth first search algorithm (BFS), etc.

The third is algorithms based on intelligent control technology, such as artificial neural network algorithms, genetic algorithms, etc.

This paper chooses A* algorithm to solve the path planning problem in the partition.

- **A* algorithm**

The A* algorithm is a heuristic algorithm that comprehensively evaluates each node of the grid by setting an evaluation function. And each node is the position reached by the robot, and each position point is intelligently evaluated to find the best position and finally find the target position. The evaluation function of the A* algorithm is as follows:

$$F_{(n)} = G_{(n)} + H_{(n)} \tag{16}$$

$F_{(n)}$ is an evaluation function of the total process node, representing the total estimated cost from the start node to the target node, $G_{(n)}$ is the actual cost from the start node to the next node in a specific state space; $H_{(n)}$ is the current The estimated value of how much cost is needed for a node to reach the target node. The key to the shortest path is to select the cost function, so it is regarded as the core problem. For the estimation heuristic method, the Manhattan estimation method is usually used.

$$H(n) = D \times (ab(n \cdot x - goal \cdot x) + ah(n \cdot y - goal \cdot y)) \tag{17}$$

Where D is the Manhattan distance, and ab is the area of the rectangle to which the bottom belongs.

The specific process of the A* algorithm is shown in the table below.

```
A* Algorithm:
Initialize open_set and close_set;
Add the starting point to open_set and set the priority to 0 (the highest priority);
If open_set is not empty, select the highest priority node n from open_set:
    If node n is the end point, then:
        Gradually track the parent node from the end point, until reaching
the starting point;
        Return the found result path, the algorithm ends;
    If node n is not the end point, then:
        Delete node n from open_set and add it to close_set;
        Traverse all neighboring nodes of node n:
            If the neighboring node m is in close_set, then:
                Skip, select the next neighboring node
            If the neighboring node m is not in the open_set, then:
                Set the parent of node m to node n
```

Compute the priority of node *m*
Add node *m* to open_set

7.1.3 Results of Zigzag parallel of the single-layer contour pattern

We use the Zigzag parallel hatch method to mark the single-layer contour pattern obtained from Attachment 1, taking d=0.1mm and d=1mm respectively. When d=1mm, the marking result is shown in Figure 17(a), when d=0.1mm, the marking result is shown in Figure 17(b).

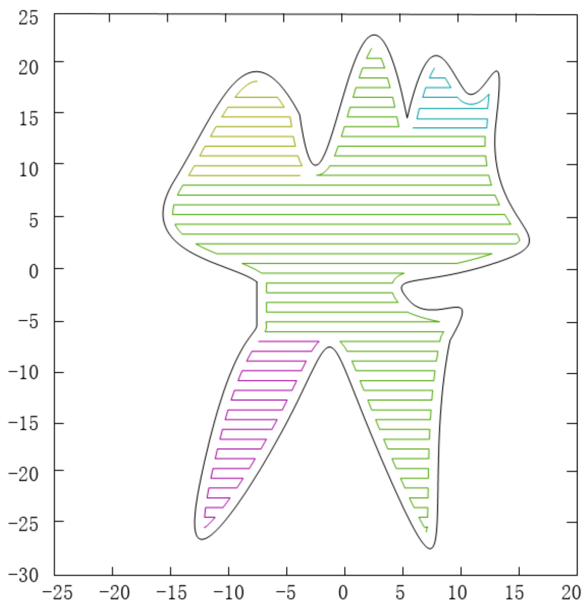
When d=1mm, we run 10 times in Matlab2018a to get the running time, and calculate the average elapsed time as shown in Table 2. Similarly, when d=0.1mm, 10 running times and average elapsed time are shown in the Table 3. According to Table 2 and Table 3, the ratio of elapsed time for program running under conditions of parameter groups (2) and (1) is 30.49.

Table 2 The average elapsed time (unit: ms) of zigzag parallel hatch of Fig.1(a) (d=1mm)

	1	2	3	4	5	Average time
Running time	319.5	332.3	321.2	326.6	324.1	324.64ms
	6	7	8	9	10	
Running time	320.6	328.9	323.1	322.8	327.3	

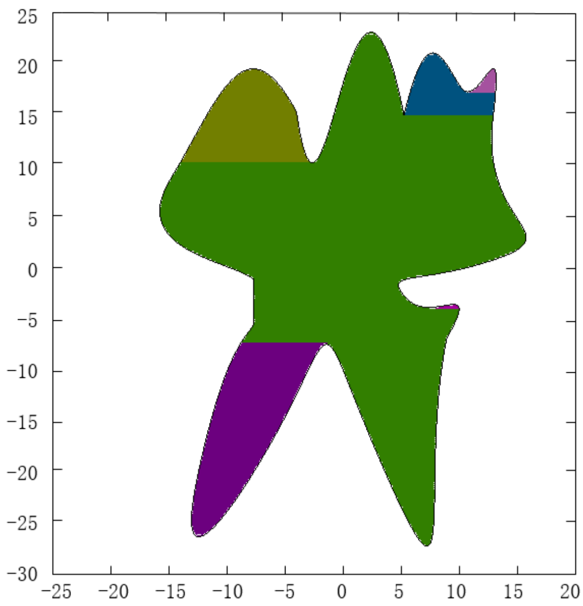
Table 3 The average elapsed time (unit: ms) of zigzag parallel hatch of Fig.1(a) (d=0.1mm)

	1	2	3	4	5	Average time
Running time	9923	9887	9896	9913	9917	9900.9ms
	6	7	8	9	10	
Running time	9903	9880	9891	9889	9910	



- Total length of hatching lines: **893.058**mm
- Number of horizontal lines:**85**
- Average elapsed time:**324.6ms**

(a)



- Total length of hatching lines: **10009.4127**mm
- Number of horizontal lines:**901**
- Average elapsed time: **9900.9ms**

(b)

Fig.17 The results of zigzag parallel hatch of single-layer contour pattern. (a) d=1mm, (b) d=0.1mm.

7.2 Contour parallel of the single-layer contour pattern

7.2.1 Path planning model

7.2.1.1 Model building

Similarly, facing Contour Parallel Hatch, path search can be subdivided into several shortest path problems. We assume that the laser executes Contour Parallel Hatch in a counterclockwise direction.

Step 1: The distance between each grid and the contour Ω is d_i. Select the grid set A where $d_i \leqslant d$ and the left grid set is A_1, and the right grid set is A_2, as shown in Figure 18(a).

Step 2: Starting from the highest point of the innermost grid P_i and ending at the lowest point of the innermost grid P_j, traverse all grids in A_1 to find the shortest path between two points, as shown in Figure 18(b).

Step 3: Starting from the lowest point of the innermost grid P_j and ending at the highest point of the innermost grid P_i, traverse all grids in A_2 to find the shortest path between two points, as shown in Figure 18(b).

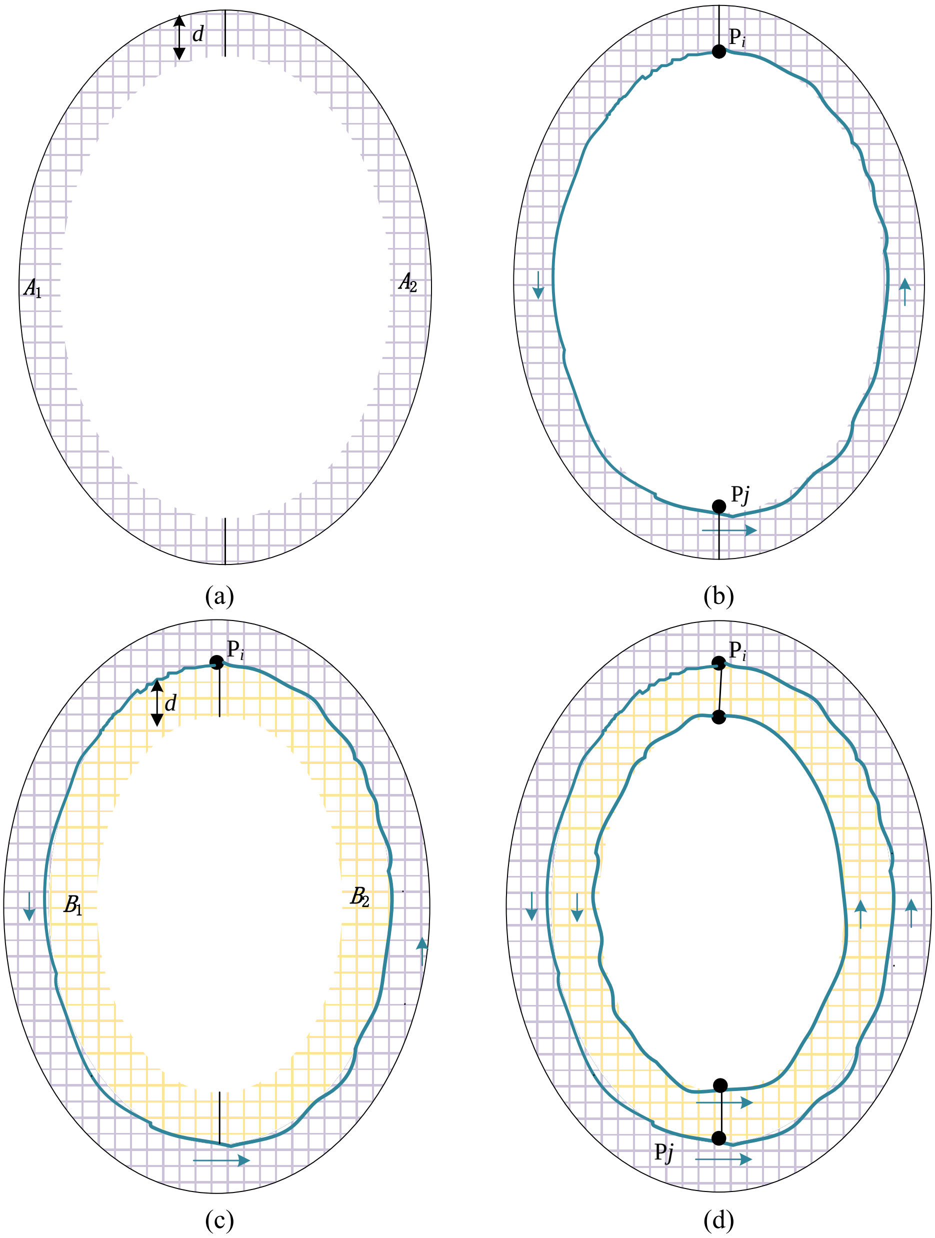
Step 4: We use the shortest path found in step 2 and step 3 to form the contour line Ω_x.The distance between each grid and the contour Ω_x is d_j. Select the grid set B where $d_j \leqslant d$ and the left grid set is B_1, and the right grid set is B_2, as shown in Figure 18(c).

Step 5: Starting from the highest point of the innermost grid P_{i+1} and ending at the lowest point of the innermost grid P_{j+1}, traverse all grids in B_1 to find the shortest path between two points, as shown in Figure 18(d).

Step 6: Starting from the lowest point of the innermost grid P_{j+1} and ending at the highest point of the innermost grid P_{i+1}, traverse all grids in B_2 to find the shortest path between two points, as shown in Figure 18(d).

Step 7: Repeat steps 2,3,4,5,6 until the distance $l_{P_{i+n}P_{j+n}}$ where the $l_{P_{i+n}P_{j+n}}$ is between the highest points of the innermost grid P_{i+n} and lowest points of the innermost grid P_{j+n}.

Step 8: End



(a) (b) (c) (d)

Fig.18 Search strategy of contour parallel hatch

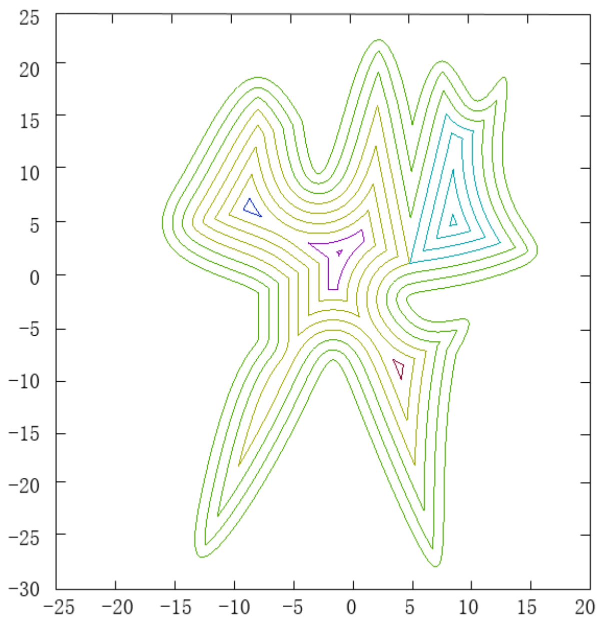
7.2.2.2 Optimization algorithm

This part is consistent with the optimization algorithm introduced in 7.1.2.2. Please refer to 7.1.2.2.

7.2.3 Results of Contour parallel hatch of the single-layer pattern

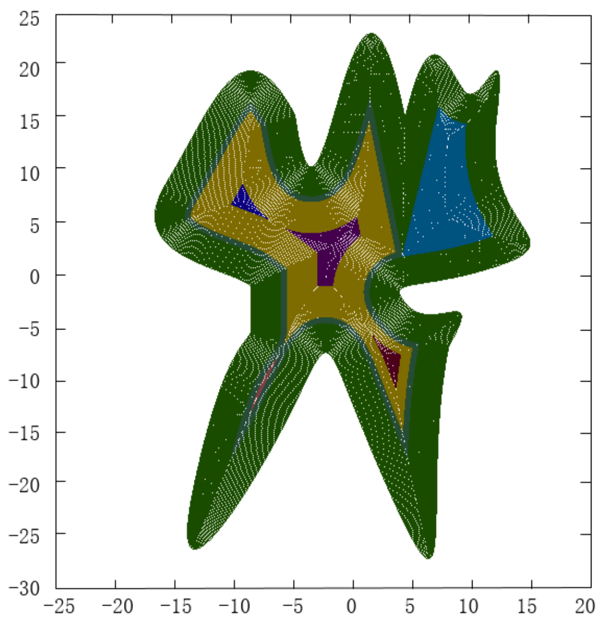
We use the Contour parallel hatch method to mark the single-layer contour pattern obtained from Attachment 1, taking *d*=0.1mm and *d*=1mm respectively. When *d*=1mm, the marking result is shown in Figure 19(a), when *d*=0.1mm, the marking result is shown in Figure 19(b).

When *d*=1mm, we run 10 times in Matlab2018a to get the running time, and calculate the average elapsed time as shown in Table 4. Similarly, when *d*=0.1mm, 10 running times and average elapsed time are shown in the Table 5. According to Table 4 and Table 5, the ratio of elapsed time for program running under conditions of parameter groups (2) and (1) is **21.98.**



- Total length of hatching lines: **895.954**mm
- Number of circles:**14**
- Average elapsed time: **819.6ms**

(a)



- Total length of hatching lines: **10017.469**mm
- Number of circles:**132**
- Average elapsed time: **18015.9ms**

(b)

Fig.19 The results of contour parallel hatch of single-layer contour pattern. (a) *d*=1mm, (b) *d*=0.1mm

Table 4 The average elapsed time (unit: ms) of contour parallel hatch of Fig.1(a) (d=1mm)

	1	2	3	4	5	Average time
Running time	813.4	815.5	820.3	821.8	822.6	819.65ms
	6	7	8	9	10	
Running time	820.9	819.3	818.6	823.6	820.5	

Table 5 The average elapsed time (unit: ms) of contour parallel hatch of Fig.1(a) (d=0.1mm)

	1	2	3	4	5	Average time
Running time	17989	17969	18008	18038	18027	18015.9ms
	6	7	8	9	10	
Running time	18043	18063	18026	17993	18003	

8 Model building and solution of question 2

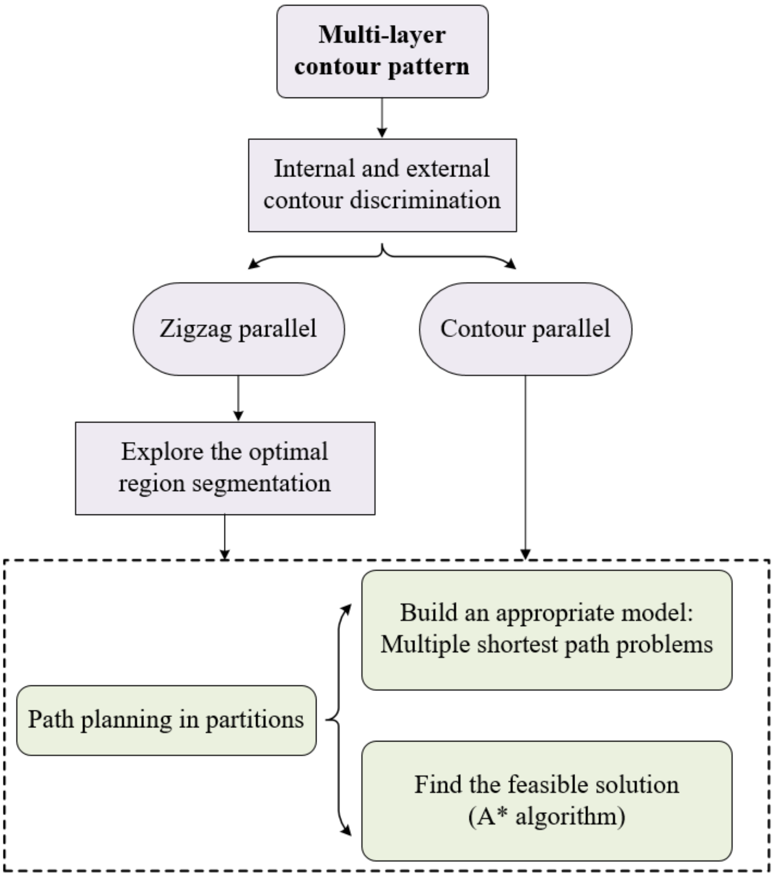


Fig.20 Flow chart of the solution of question 2

8.1 Zigzag parallel hatch of the multi-layer contour pattern

8.1.1 Regional Convex segmentation

According to the regional convex segmentation method introduced in 7.1.1, the partition map of the multi-layer contour pattern in Attachment 2 can be obtained, as shown in Figure 21 below. There are 7 partitions in the multi-layer contour pattern drawn according to Attachment 2.

8.1.2 Path planning model in partitions

8.1.2.1 Discrimination of inner and outer contours

As we can see from the data in Attachment 2, the problem we want to solve is the hatch scanning problem in the multi-layer contour patterns. Before performing the Zigzag Parallel and Contour Parallel hatch, we perform the inner and outer contour discrimination of the nested multi-layer contour pattern.

In geometry, the concept of a ring is the boundary of a surface, in which the contour ring is closed and the region enclosed by the contour ring is a surface. The nested multi-layer contour pattern can cause the inner and outer rings to be included or separated.

The judgment rule is as follows: for the nested multi-layer contour pattern, we take any two contour rings, we take a certain point of a ring, and make horizontal ray from this point to the left or right. At this time, the parity of the intersection point of this ray and another ring is the key to determine its position relationship. If odd, then the first ring is contained by the second ring. If it's even, let's go ahead and take a point from the second ring and make a horizontal ray to the left or right. If it intersects the first ring with an odd number, then the second ring contains the first ring. If its intersection with the first ring is even, the rings are separated.

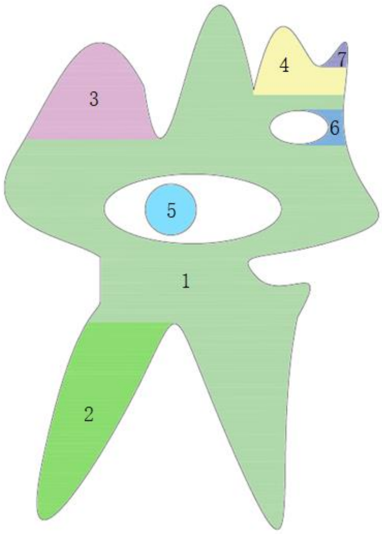


Fig.21 Partition map of multi-layer pattern drawn according to Attachment 2

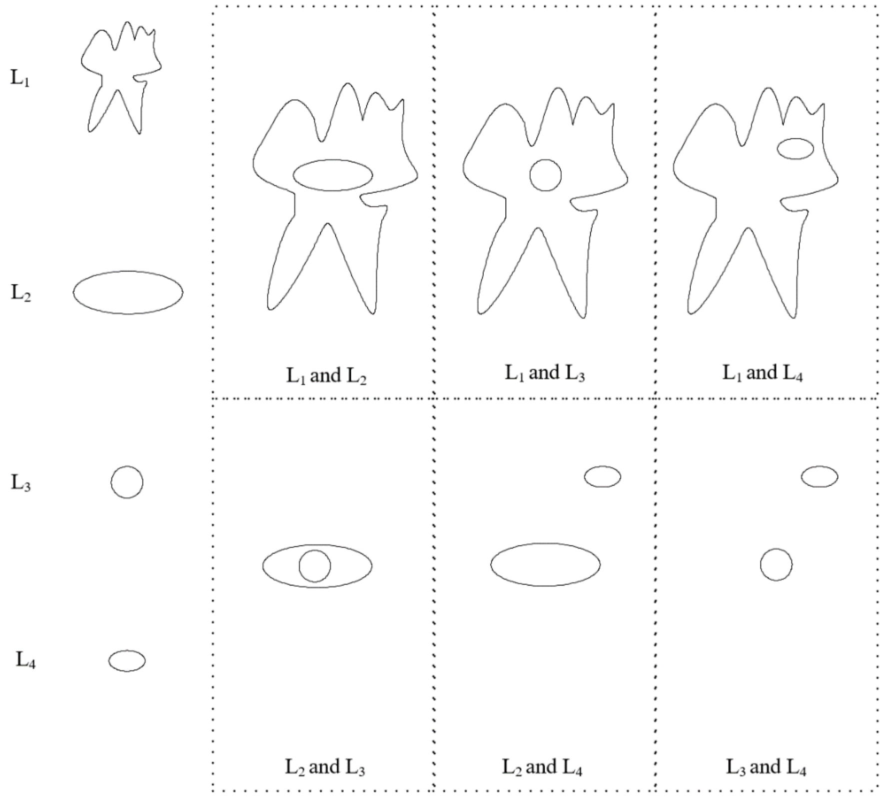


Fig.22 The internal and external contour analysis for multilayer contour pattern in Attachment 2

As shown in Figure 22, we disassembled the multi-layer contour pattern into six groups, they are $L_1, (L_1, L_3), (L_1, L_4), L_3, (L_2, L_4)$ and (L_3, L_4). We can get the nested relationships from the six contour combinations as follows: L_2 is in L_1, L_3 is in L_2, L_4 is in L_1, L_4 is out L_2.

So, the area to be marked is $(L_1 \cap L_2) \cup L_3$.

8.1.2.2 Model building

This part is consistent with the model building process introduced in 7.1.2.1. Please refer to 7.1.2.1.

8.1.2.3 Optimization algorithm

This part is consistent with the optimization algorithm introduced in 8.1.2.2. Please refer to 7.1.2.2.

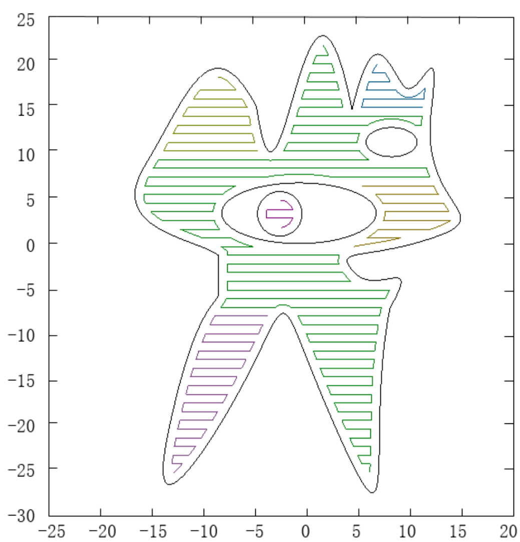
8.1.3 Results of Zigzag parallel of the multi-layer contour pattern

We use the Zigzag Parallel hatch method to mark the multi-layer contour pattern obtained from Attachment 2, taking *d*=0.1mm and *d*=1mm respectively. When *d*=1mm, the marking result is shown in Figure 23(a), when *d*=0.1mm, the marking result is shown in Figure 23(b).

When *d*=1mm, we run 10 times in Matlab2018a to get the running time, and calculate the average elapsed time as shown in Table 6. Similarly, when *d*=0.1mm, 10 running times and average elapsed time are shown in the Table 7. According to Table 6 and Table 7, the ratio of elapsed time for program running under conditions of parameter groups (2) and (1) is **7.19**.

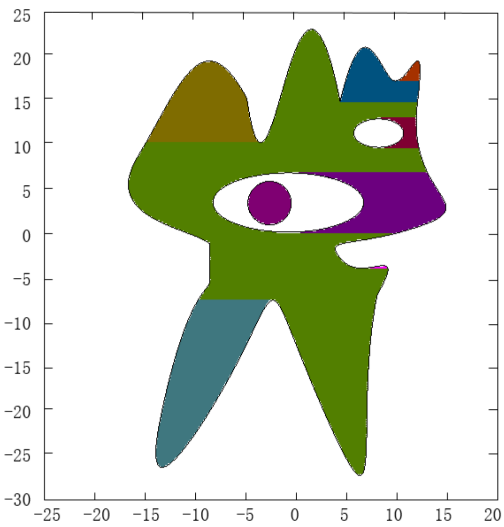
Table 6 The average elapsed time (unit: ms) of zigzag parallel hatch of Fig.1(b) (*d*=1mm)

	1	2	3	4	5	Average time
Running time	1508	1516	1503	1498	1486	1503.5ms
	6	7	8	9	10	
Running time	1493	1520	1515	1502	1494	



(a)

- ✧ Total length of hatching lines: **742.842**mm
- ✧ Number of horizontal lines:**89**
- ✧ Average elapsed time:**1503.5ms**



- ✧ Total length of hatching lines: **9049.608**mm
- ✧ Number of horizontal lines:**1056**
- ✧ Average elapsed time: **10816.5ms**

(b)

Fig.23 The results of zigzag parallel hatch of multi-layer contour pattern. (a) *d*=1mm, (b) *d*=0.1mm

Table 7 The average elapsed time (unit: ms) of zigzag parallel hatch of Fig.1(b) (*d*=0.1mm)

	1	2	3	4	5	Average time
Running time	10795	10758	10769	10837	10829	10816.5ms
	6	7	8	9	10	
Running time	10842	10777	10842	10854	10862	

8.2 Contour parallel hatch of the multi-layer contour pattern

8.2.1 Path planning model

8.2.1.1 Model building

This part is consistent with the model building process introduced in 7.2.2.1. Please refer to 7.2.2.1.

8.2.1.2 Optimization algorithm

This part is consistent with the optimization algorithm introduced in 7.2.2.2. Please refer to 7.2.2.2.

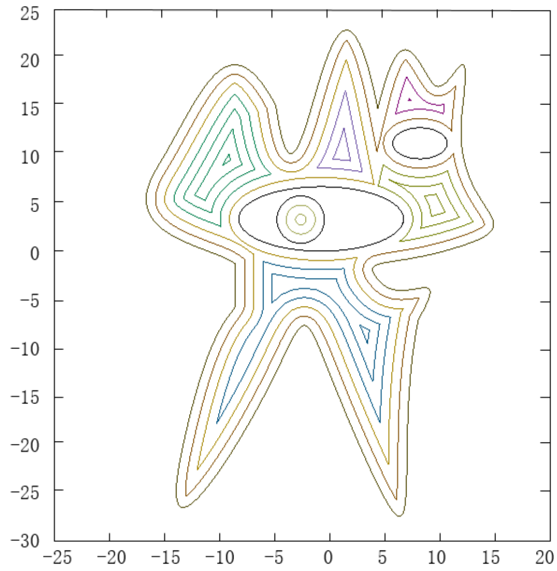
8.2.2 Results of Contour parallel hatch of the multi-layer pattern

We use the Contour Parallel hatch method to mark the multi-layer contour pattern obtained from Attachment 2, taking *d*=0.1mm and *d*=1mm respectively. When *d*=1mm, the marking result is shown in Figure 24(a), when *d*=0.1mm, the marking result is shown in Figure 24(b).

When *d*=1mm, we run 10 times in Matlab2018a to get the running time, and calculate the average elapsed time as shown in Table 8. Similarly, when *d*=0.1mm, 10 running times and average elapsed time are shown in the Table 9. According to Table 8 and Table 9, the ratio of elapsed time for program running under conditions of parameter groups (2) and (1) is **8.48**。

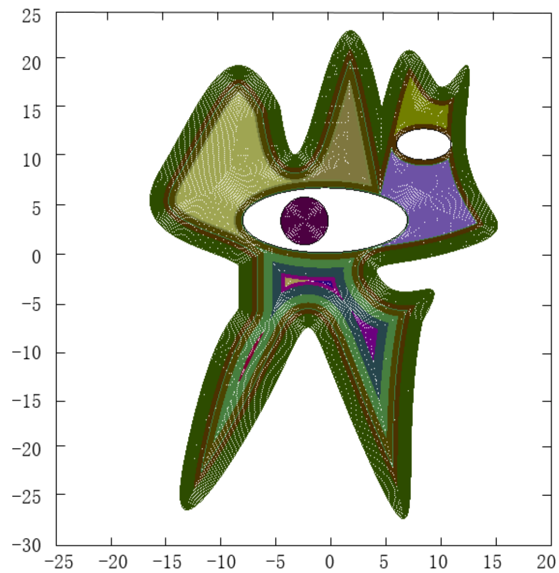
Table 8 The average elapsed time (unit: ms) of contour parallel hatch of Fig.1(b) (d=1mm)

	1	2	3	4	5	Average time
Running time	2710	2734	2755	2673	2688	2712.3ms
	6	7	8	9	10	
Running time	2692	2697	2744	2722	2708	



(a)

- ✧ Total length of hatching lines: **766.781**mm
- ✧ Number of circles:**18**
- ✧ Average elapsed time: **2712.3ms**



(b)

- ✧ Total length of hatching lines: **9056.616**mm
- ✧ Number of circles:**198**
- ✧ Average elapsed time: **23007.4ms**

Fig.24 The results of contour parallel hatch of multi-layer contour pattern. (a) d=1mm, (b) d=0.1mm

Table 9 The average elapsed time (unit: ms) of contour parallel hatch of Fig.1(b) (d=0.1mm)

	1	2	3	4	5	Average time
Running time	22967	22933	23087	23076	22959	23007.4ms
	6	7	8	9	10	
Running time	23015	22976	23024	23041	22996	

9 Question 3: Optimization strategy

When we optimize the efficiency and performance of the hatching mathematical model, we mainly consider from the following two aspects:

- When it comes to the goal of overall optimization, we should comprehensively measure the optimization of three aspects, namely, regional segmentation mode, internal scanning planning of sub-regions, and cohesion order of sub-regions.
  In addition, we should give special consideration to the separate optimization of the cohesion order of all subregions. We regard the cohesion order problem of multiple subregions as a multi-travel agent (TSP) problem, and we suggest using Hopfield neural network algorithm to solve and optimize the cohesion order between subregions [6].
- We optimize the internal hatch scan planning algorithm of the subregion. We consider using artificial potential field [7] method combined with A* algorithm to carry out hatch scanning path planning promotion. In this way, not only A* algorithm can escape from local minima quickly, but also the traditional artificial potential field method can be used to solve the problem of relatively low real-time performance of A* algorithm.

10 Strength and weakness

10.1 Strengths

(1) We have tried a variety of partitioning strategies and carried out parallel strategy scanning based on the optimal partitioning to ensure the model effect.

(2) We use multiple path search algorithms to modify the laser scanning strategy, which greatly improves the accuracy and reliability of the model.

(3) The model has good explanatory ability, and the process-based model better explains the details in the laser scanning shadow process.

10.2 Weaknesses

(1) Although we consulted the materials and learned the knowledge of filling shadows by laser hatch scanning, the time is limited and our understanding is relatively shallow. Fitting the external closed contour according to the attached data is a simplified operation, which may bring errors to the results.

(2) Facing the irregular contour, the scanning requirements are as few turns as possible and the total path is short. However, our overall model is difficult to take into account the multi-optimization objectives, and there are still many improvements to be made in details.

Reference

[1] B X S A , B W W A , B X M A , et al. Femtosecond laser dot-matrix marking on nickel-based alloy using a simple diaphragm-based spatial shaped modulation: Size and position control of marking units with high recognition rate[J]. Journal of Alloys and Compounds, 835.
[2] Lu G , Trdan U , Zhang Y , et al. The distribution regularity of residual stress on a metal surface after laser shock marking[J]. Mechanics of Materials, 2020, 143:103310-.
[3] Gibson I , Rosen D , Stucker B . Additive manufacturing technologies: 3D printing, rapid prototyping, and direct digital manufacturing, second edition[M]. Springer US, 2015.
[4] Latombe J C. Exact cell segmentation Robot Motion Planning[M]. Springer US,1991:200-247.
[5] Oksanen T, Visala A. Coverage Path Planning algorithms for agricultural field machines [J]. Journal of Field Robotics,2007,26(8):651-668.
[6] Wang JianZhao, HemingXiao, Jinqiu. Optimized Region Decomposition Based Complete Coverage Path Planning for Mobile Robot [J]. Computer Engineering and applications ,2006(22):59-62.
[7] Yao Yuan, Zhou Xing-she, Zhang Kai-long, Dong Dong. Dynamic trajectory planning for unmanned aerial vehicle based on sparse A* search and improved artificial potential field [J]. Control Theory & Applications,2010,27(07):953-959.

Appendix

Generate Zigzag scan order

```
%Run in the MATLAB2018a
clear all;
clc;

function matrix = ZigZag(n)
    matrix = zeros(n);
    counter = 1;
    flipCol = true;
    flipRow = false;

    %This for loop does the top-diagonal of the matrix
    for i = (2:n)
        row = (1:i);
        column = (1:i);

        %Causes the zig-zagging.
        if flipCol
            column = fliplr(column);
            flipRow = true;
```

```
            flipCol = false;
        elseif flipRow
            row = fliplr(row);
            flipRow = false;
            flipCol = true;
        end

        %Selects a diagonal of the zig-zag matrix and places the correct
        %integer value in each index along that diagonal
        for j = (1:numel(row))
            matrix(row(j),column(j)) = counter;
            counter = counter + 1;
        end
    end

    %This for loop does the bottom-diagonal of the matrix
    for i = (2:n)
        row = (i:n);
        column = (i:n);

        %Causes the zig-zagging.
        if flipCol
            column = fliplr(column);
            flipRow = true;
            flipCol = false;
        elseif flipRow
            row = fliplr(row);
            flipRow = false;
            flipCol = true;
        end

        %Selects a diagonal of the zig-zag matrix and places the correct
        %integer value in each index along that diagonal
        for j = (1:numel(row))
            matrix(row(j),column(j)) = counter;
            counter = counter + 1;
        end
    end

end
```

Generate the shortest path
clear all; clc;

```
[m n]=size(M1);
[x y]=ginput();
x=round(x);
y=round(y);

tmp=ones(m,n);
queue_head=1;
queue_tail=1;
neighbour=[-1 -1;-1 0;-1 1;0 -1;0 1;1 -1;1 0;1 1];
%neighbour=[-1 0;1 0;0 1;0 -1];
q{queue_tail}=[y x];
queue_tail=queue_tail+1;
[ser1 ser2]=size(neighbour);

while queue_head~=queue_tail
    pix=q{queue_head};
    for i=1:ser1
        pix1=pix+neighbour(i,:);
        if pix1(1)>=1 && pix1(2)>=1 &&pix1(1)<=m && pix1(2)<=n
            if img(pix1(1),pix1(2))==1
                img(pix1(1),pix1(2))=0;
                q{queue_tail}=[pix1(1) pix1(2)];
                queue_tail=queue_tail+1;

            end
        end
    end

    queue_head=queue_head+1;
end

figure(1);
imshow(mat2gray(M1));
```

Isometric contraction

```
public static Point2D[] Expand(Point2D[] polygon, double expand)
{
Point2D[] new_polygon = new Point2D[polygon.Length];
int len = polygon.Length;
for (int i = 0; i < len; i++)
{
Point2D p = polygon[i];
Point2D p1 = polygon[i == 0 ? len - 1 : i - 1];
Point2D p2 = polygon[i == len - 1 ? 0 : i + 1];
double v1x = p1.X - p.X;
```

```
double v1y = p1.Y - p.Y;
double n1 = norm(v1x, v1y);
v1x /= n1;
v1y /= n1;
double v2x = p2.X - p.X;
double v2y = p2.Y - p.Y;
double n2 = norm(v2x, v2y);
v2x /= n2;
v2y /= n2;
double l = -expand / Math.Sqrt((1 - (v1x * v2x + v1y * v2y)) / 2);
double vx = v1x + v2x;
double vy = v1y + v2y;
double n = l / norm(vx, vy);
vx *= n;
vy *= n;
new_polygon[i] = new Point2D(vx + p.X, vy + p.Y);
}
return new_polygon;
}
private static double norm(double x, double y)
{
return Math.Sqrt(x * x + y * y);
}
```