Path Planning for Force-controlled Robotic Grinding of Hub Surfaces

Yixiao Wang

ABSTRACT

The production of aluminum alloy wheels has been basically automated, but the grinding process is still manual. Therefore, robotic grinding has gradually attracted the attention from academia and industry, and researchers mainly focus on the design of the grinding actuator and the research of the grinding mechanism. As an important part of robot grinding, path planning is one of the determinants of grinding effect and efficiency and needs more research.

This dissertation takes force-controlled robotic grinding of the wheel hub surface as the object and studies its path planning algorithm. The main work is listed as follows:

Aiming at the problem that the direct path planning on the complex surface will cause the path to be interrupted and make the grinding tool leave the surface of the workpiece, this paper proposes a surface segmentation algorithm based on sIeping lines, so that any path composed of parallel sIeping lines is guaranteed be continuous in any sub-surface. Morse decomposition based on specified function pattern is a common algorithm to do the same job in full coverage path planning. Inspired by this, this paper proposes a surface segmentation algorithm based on sIeping lines. First, digitizes the function pattern, and then clusters the sIeping lines by judging the overlap of adjacent sIeping lines. Each cluster represents one final sub-surface.

Aiming at the multi-objective local path planning problem of simultaneously optimizing the grinding effect and efficiency, this paper proposes a greedy algorithm by continuously solving constrained optimization problems and a global algorithm by solving single large-scale optimization problem to plan the feed rate and path spacing. First, according to the grinding mechanics model, the material removal rate per unit time on the surface of the workpiece is calculated through the grinding force and related parameters. Then, establish a constrained nonlinear optimization problem with the goal of maximizing the total grinding volume based on discretized single grinding path points and solve it to plan the feed rate of this grinding path. Finally, based on the feed rate planning model and results, the greedy algorithm is used to plan the distance from the next grinding path; or discretize multiple grinding path, and optimize the multiple path spacings and corresponding feed rates at the same time with the goal of minimizing the average residual volume to be polished.

Aiming at the problem of grinding interval movement planning, that is, the problem of non-grinding path planning, this paper constructs an integer programming problem that takes the result of solving the traveling salesman problem as the objective function, and then optimize the types of local paths and the order of visiting all the sub-surfaces with the goal of minimizing the total length of interval movement.

KEY WORDS: wheel hub surface, force-controlled grinding, surface segmentation, grinding mechanism, local path planning, interval movement planning

Numerical Surface Segmentation Based on Sweeping Lines

Problem statement

Target area to be ground is upper surface of the automobile hub, shown as Figure 1-1 . The problem is to divide the complex upper surface into sub surfaces so that paths made of parallel sIeping lines can be continuous in each sub surface. The output in Figure 1 shows in each subsurface whose boundary is drawn as the same color, paths made of parallel arcs can be continuous.

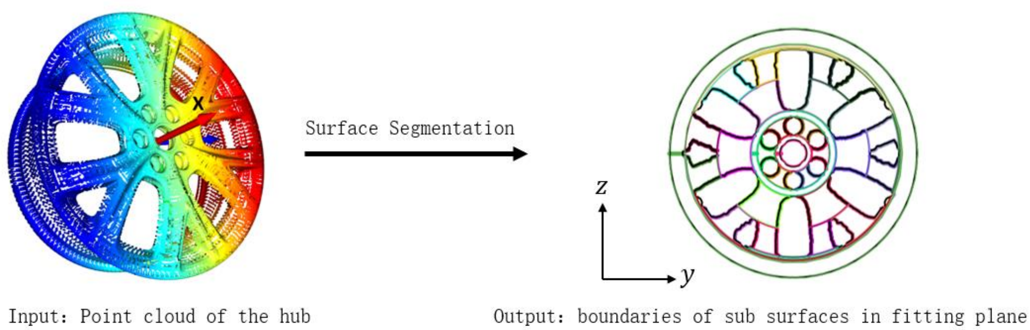


Figure 1-1 Illustration of the input and output of proposed surface segmentation algorithm

Algorithm design

First, I extract the upper surface of the hub, shown as Figure 1-2 and Figure 1-3.

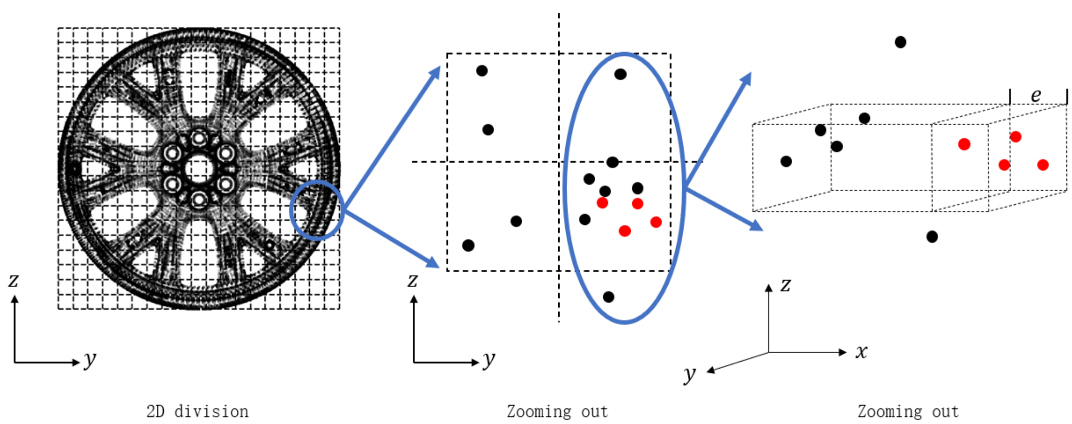


Figure 1-2 Illustration of the algorithm to extract the upper surface of automobile hub

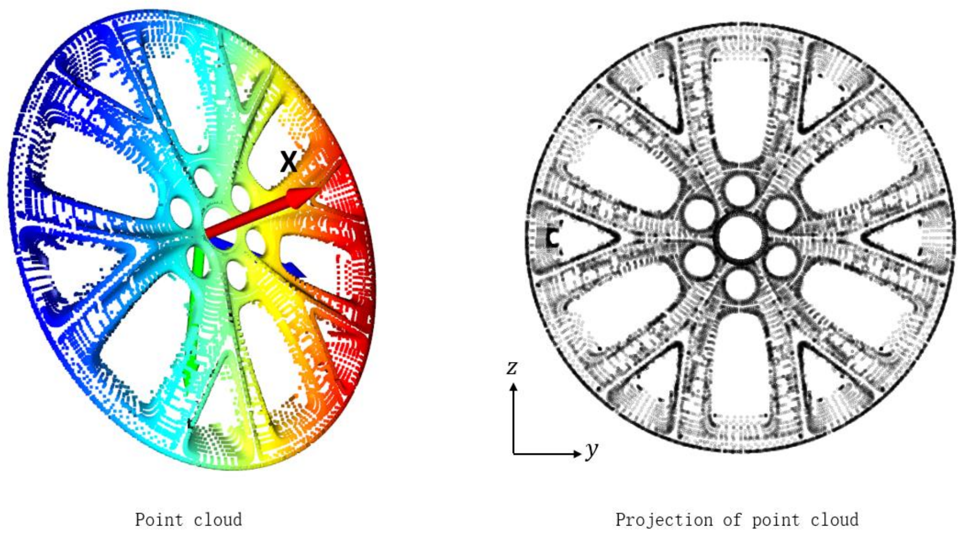


Figure 1-3 Illustration of upper surface of automobile hub through proposed method

Second, I detect the boundary through modified AlphaShape algorithm. I establish Delaunay triangulation, as shown in Figure 1-4. Then I delete several bad triangles according to some standards inspired by AlphaShape algorithm. Finally, I modify depth first search algorithm and detect the boundary. The algorithm is shown in Table 1-1. And the final result is shown in Figure 1-5. The computational complexity of the modified AlphaShape is $O(Nlog(N))$, where N is the number of points. And the computational complexity of traditional one is $O(N^3)$. That shows modified AlphaShape is very suitable to deal with a large number of point clouds.

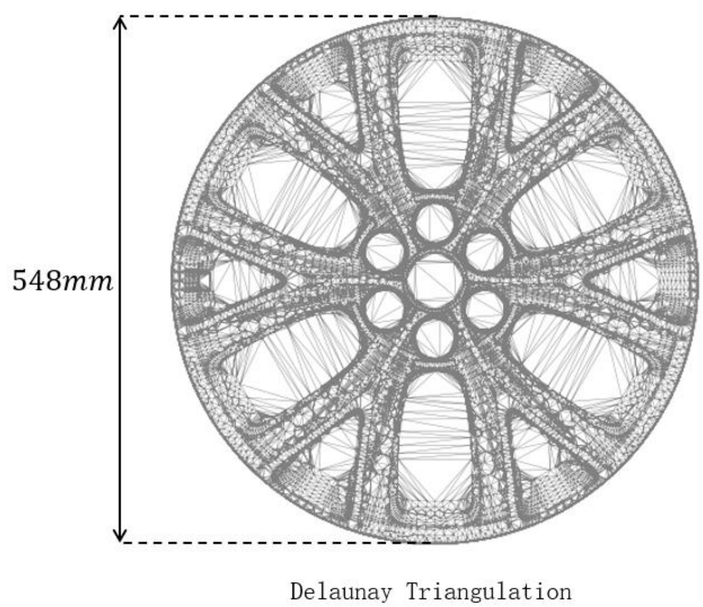


Figure 1-4 Delaunay Triangulation

Table 1-1 DFS2 Algorithm

Algorithm 1: DFS2

Input: $2 \times n$ Array L, dictionary dic. Each column in L consists of two indice representing two end points of the borderline; The key in dic is one end point index and the value is the other end point index.

Output: List C. Each element in C consists of a set of contours represented by ordered point indice.

```
Initialize C ← [], visited ← set();
Function dfs2(dic,s,f,C_t = []):
    C_t ← C_t + [s];
    if len(C_t) > 1 and s == f then
        return [C_t];
    end
    visited.add(s);
    C_ts ← [];
    foreach node in dic[s] do
        if node not in C_t[1 :] then
            newC_ts ← dfs2(dic,node,f,C_t);
            foreach newC_t in newC_ts do
                C_ts.append(newC_t);
            end
        end
    end
end
for i = 0; i <L.shape[1]; i + + do
    elem ← C[:, i];
    if elem[0] not in visited then
        temp ← dfs2(dic,elem[0],elem[0]);
        flag ← 1;
        foreach t in temp do
            if len(t) > n_min then
                flag ← 0;
                break;
            end
        end
        if flag == 1 then
            foreach t in temp do
                foreach tt in t do
                    if tt in visited then
                        visited.remove(tt);
                    end
                end
            end
        end
        C.append(temp)
    end
end
```

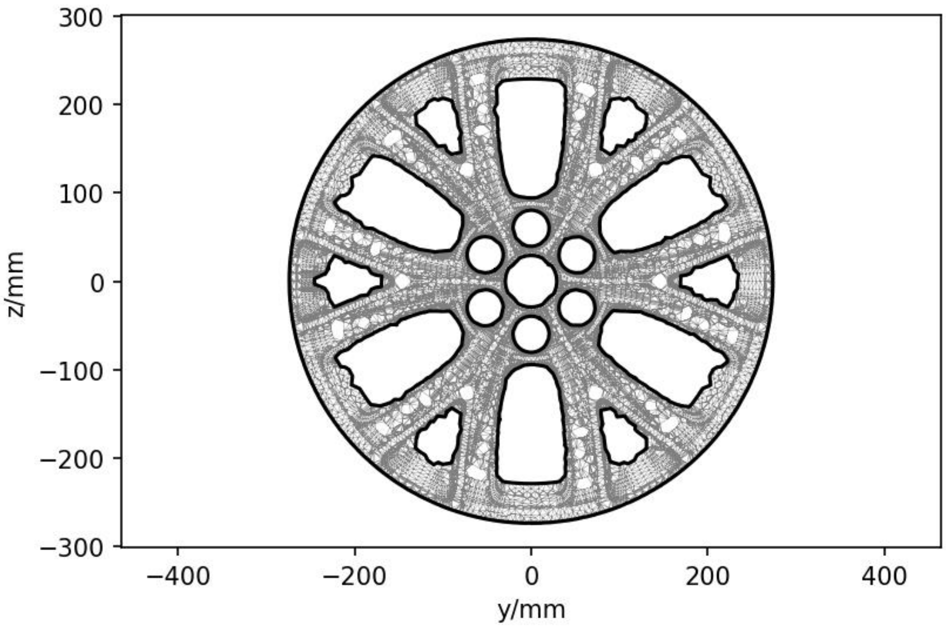


Figure 1-5 Results from modified AlphaShape algorithm

Third, I design a numerical surface segmentation algorithm based on sIeping lines. Figure 1-6 shows the result. And Figure 1-7 shows an example of the local path. I can see the parallel paths can be continuous in each sub surface.

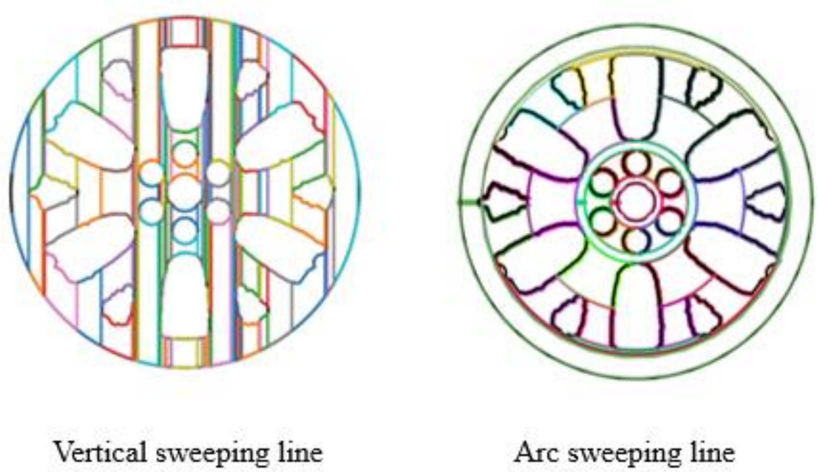
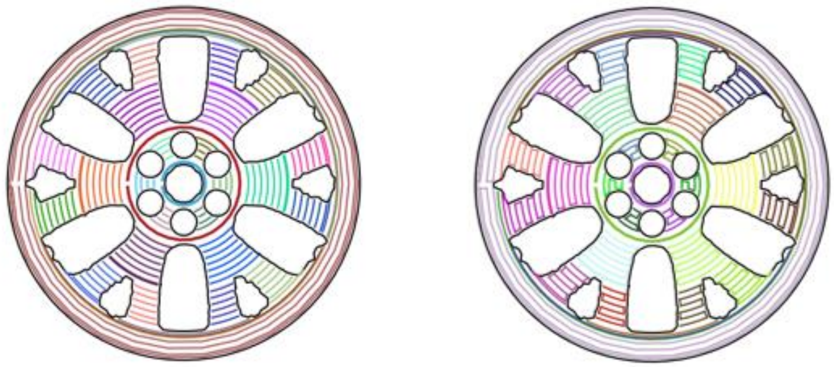


Figure 1-6 Results from two different surface segmentation



(a)local parallel paths (b)an example of local continuous paths

Figure 1-7 Examples of local paths

Local path planning of grinding

Problem Statement

Figure2-1 shows the geometry model of disk grinding. And the problem is to plan the local path in each sub surface. To be specific, the local path consists of parallel sIeping lines. The feed velocity in each sIeping line will be planned (Velocity can be different in different part of the line) and the line spacing will be planned.

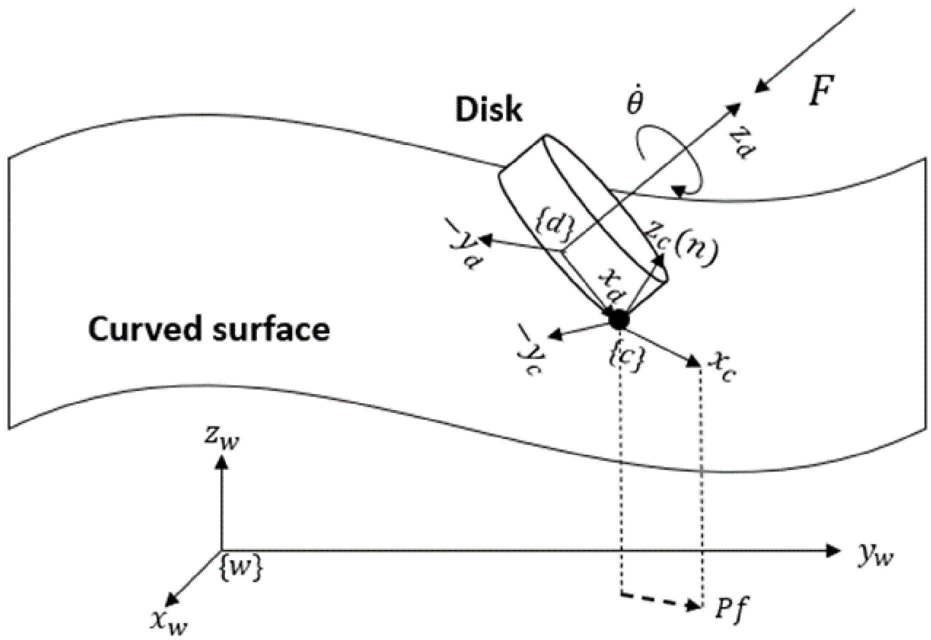


Figure 2-1 Model of disk grinding and polishing of curved surface

Algorithm Design

First, I introduce mechanics of contact force and material removal model.

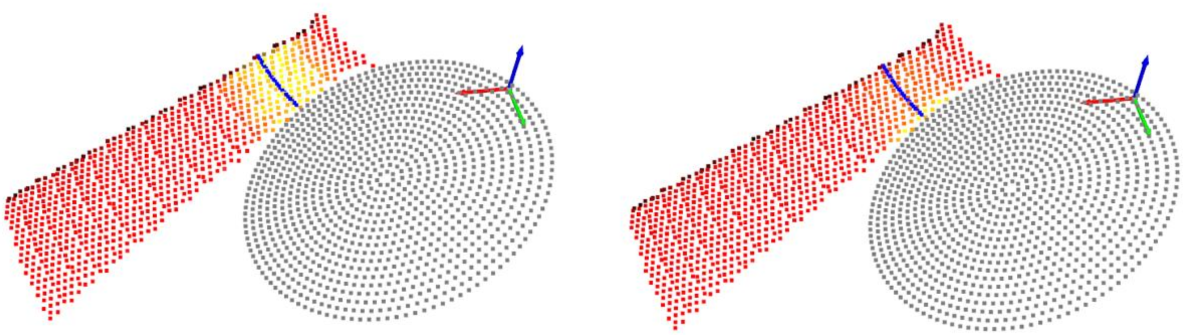
Second, I plan the feed velocity with the aim of maximizing the total removal volume. The constraints are maximum feed velocity and acceleration. Besides, there is no overground area. I discretize the path into points and convert velocity v_i planning into resident time t_i planning. The optimization problem is shown as:

$$
\begin{aligned}
&\min_{\delta t_i \in R} - \sum_i \delta t_i \sum_j w_{i,j} \\
&s.t. \sum_i w_{i,j} \delta t_i \le Re_j, \forall j \in \{0,1,2,\ldots,m\} \\
&\delta t_i \ge \frac{\delta s}{V_{max}}, \forall i \in \{0,1,2,\ldots,n\} \\
&-A_{max} \le \frac{\delta t_{i-1} - \delta t_{i+1}}{2\delta t_i^3} \delta s \le A_{max}, \forall i \in \{1,2,3\ldots,n-1\} \\
&-A_{max} \le \frac{\delta t_0 - \delta t_1}{\delta t_i^3} \delta s \le A_{max} \\
&-A_{max} \le \frac{\delta t_{n-1} - \delta t_n}{\delta t_i^3} \delta s \le A_{max}
\end{aligned}
$$

Third, I plan the path spacing through greedy strategy and global strategy.

Simulation Experiment

First, I did an experiment to show whether feed velocity planning matters. The result is shown in Figure 2-2 and Figure 2-3. The red points are surface to be ground. The colors from red to yellow mean the removal depth to be ground from initial value to zero. Blue points denote path points and grey points denote the bottom surface of the disk. The treatment group is the result after proposed feed velocity planning algorithm. And the control group is the result when the feed velocity in the line should be constant. The result shows it is vital to plan the feed velocity in hub grinding. The reason for the result of control group is that there is large repeated grinding area.



Treatment Group

Control Group

Figure 2-2 Results of simulation experiments of feed velocity planning

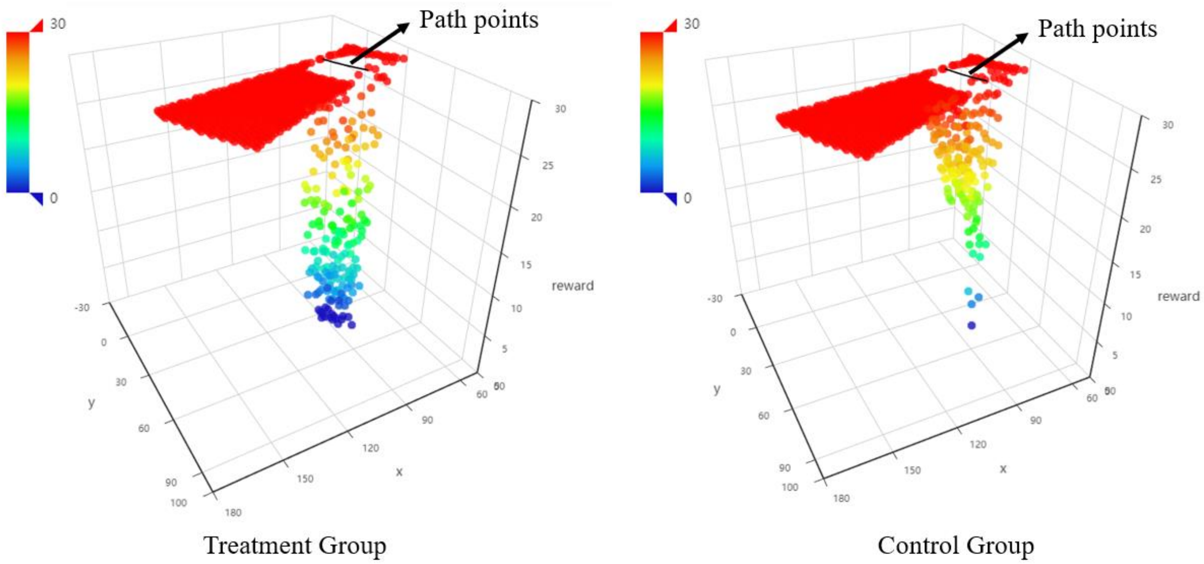


Figure 2-3 Results of simulation experiments of feed velocity planning

Second, I plan the path spacing through greedy strategy using modified pattern search algorithm as shown in Table 2-1. The result is quite nice, as shown in Figure 2-4.

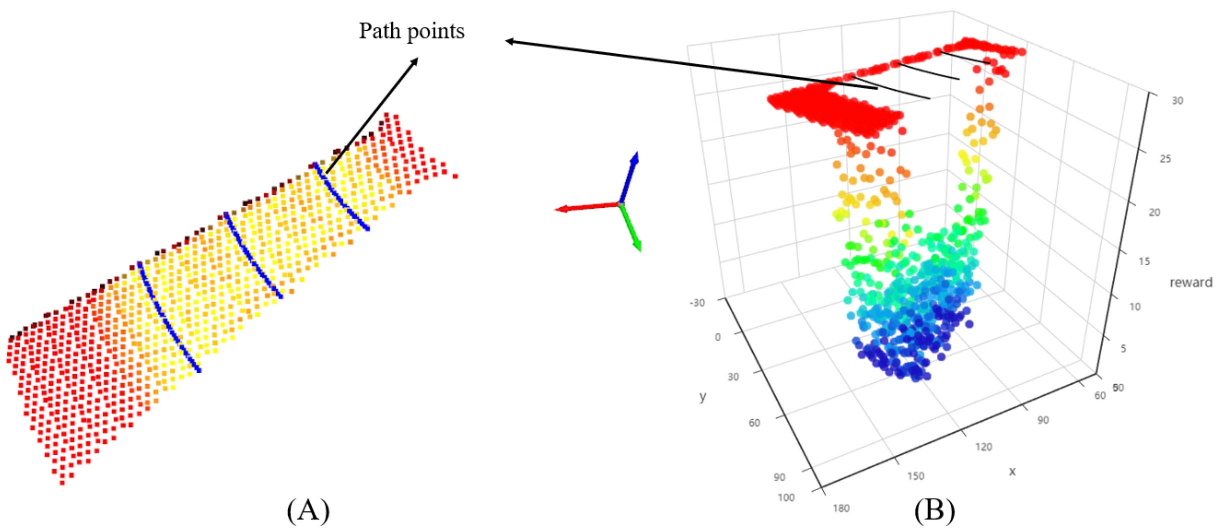


Figure 2-4 Results of path spacing planning

Table 2-1 Modified Pattern Search Algorithm

Algorithm 2: Modified Pattern Search algorithm

Input: float x_0, initial value of the variable; float *step*, initial search step size ; float $\alpha < 1$, step adjustment coefficient; float ϵ, the tolerance of search step to terminate.

Output: float x, variable value where the function value is minimized.

```
Initialize x ← x_0, x_1 ← x_0, f_min ← f(x_0);
WhetherFoundA ← False;
WhetherFoundS ← True;
while do
    WhetherFoundA ← False;
    while do
        x_1 ← x + step;
        if x_1 in Boundary then
            if f(x_1) < f_min then
                x ← x_1;
                f_min ← f(x_1);
                WhetherFoundA ← True;
                Continue;
            end
        end
        Break;
    end
    if not (WhetherFoundA == False and WhetherFoundS == True) then
        step ← step · α;
        if step ≤ ϵ then
            Break;
        end
    end
    WhetherFoundS ← False;
    while do
        x_1 ← x − step;
        if x_1 in Boundary then
            if f(x_1) < f_min then
                x ← x_1;
                f_min ← f(x_1);
                WhetherFoundS ← True;
                Continue;
            end
        end
        Break;
    end
    if not (WhetherFoundS == False and WhetherFoundA == True) then
        step ← step · α;
        if step ≤ ϵ then
            Break;
        end
    end
end
return x;
```

Interval movement planning

Problem Statement

The problem is to plan the types of local paths and visit order of all the sub surfaces, as shown in Figure 4-1.

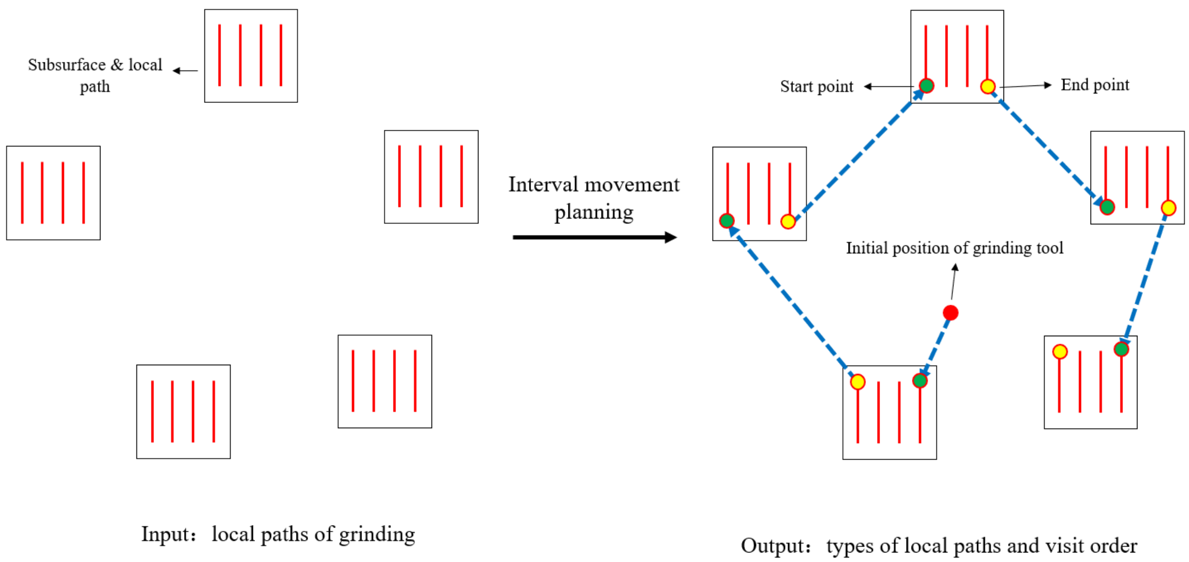


Figure 4-1 Illustration of the input and output of interval movement planning

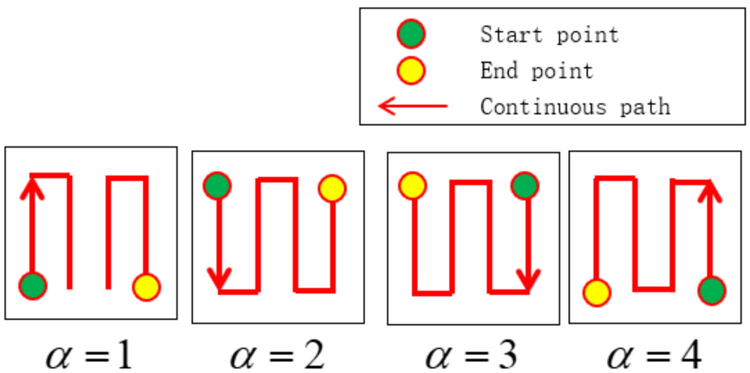


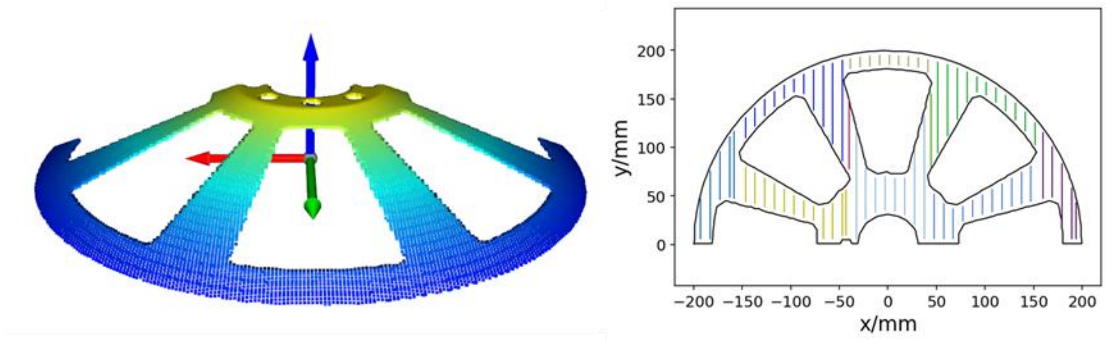
Figure 4-2 Four different types of continuous local paths

Algorithm Design

I solve this problem through genetic algorithm and LKH methods.

Simulation Experiment

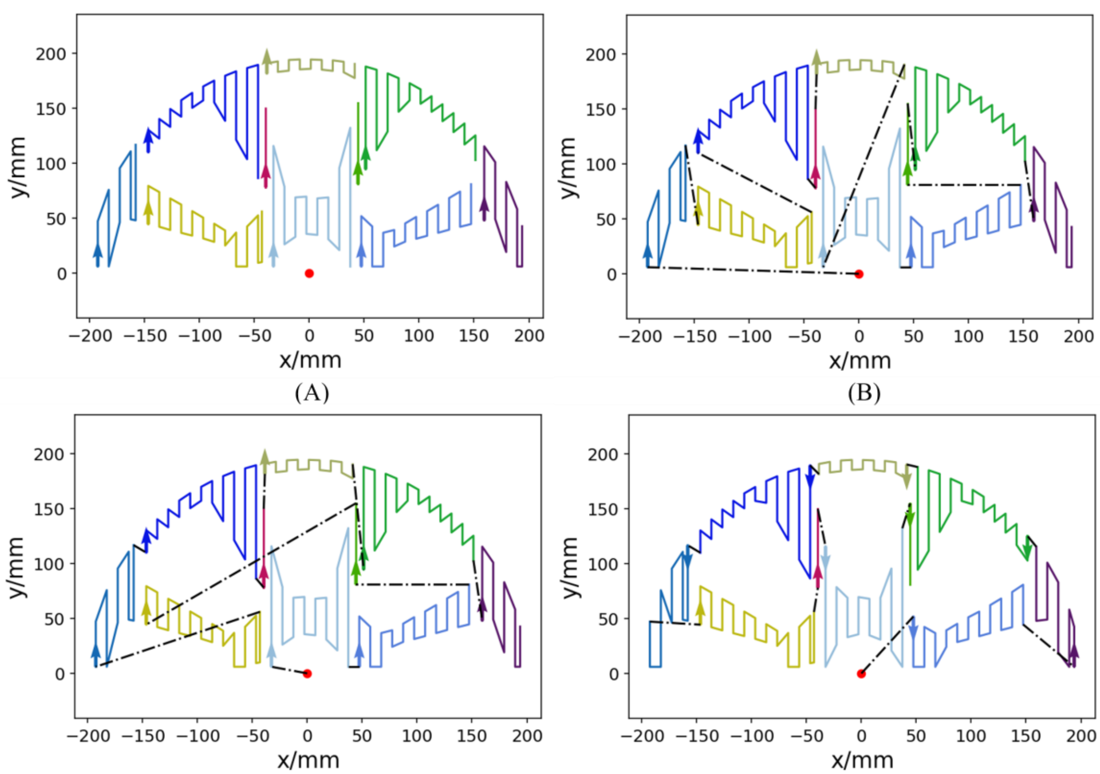
Figure 4-3 shows the object of the experiment. (A) shows the point clouds and (B) shows their projection. And the local parallel paths in each sub surface are drawn as the same color.



(A) (B)

Figure 4-3 Experiment objects of interval movement planning

Figure 4-4 shows the results of experiments. (A) shows an example of local path types. Red point denotes the initial position of grinding tool. Arrows denote the feed direction. (B) shows an example of interval movement without any planning. Black dashed lines denote interval movement. (C) shows the result after visit order planning. (D) shows the result after total interval movement planning. Table 4-1 shows the total lengths of different results. I can tell interval movement planning can significantly decrease the total length of interval movement path and reduce the times of lifting tool from the workpiece because of meeting the boundary.



(A) (B) (C) (D)

Figure 4-4 Results of simulation experiments of interval movement planning

Table 4-1 Statistics of total lengths of interval movements in simulation experiments

Interval movement	B	C	D
Total length(mm)	853	732	305

Physical Experiment

Figure 5-1 shows the environment of physical experiment. I use ATOS Triple Scan to calculate the removal depth after grinding.

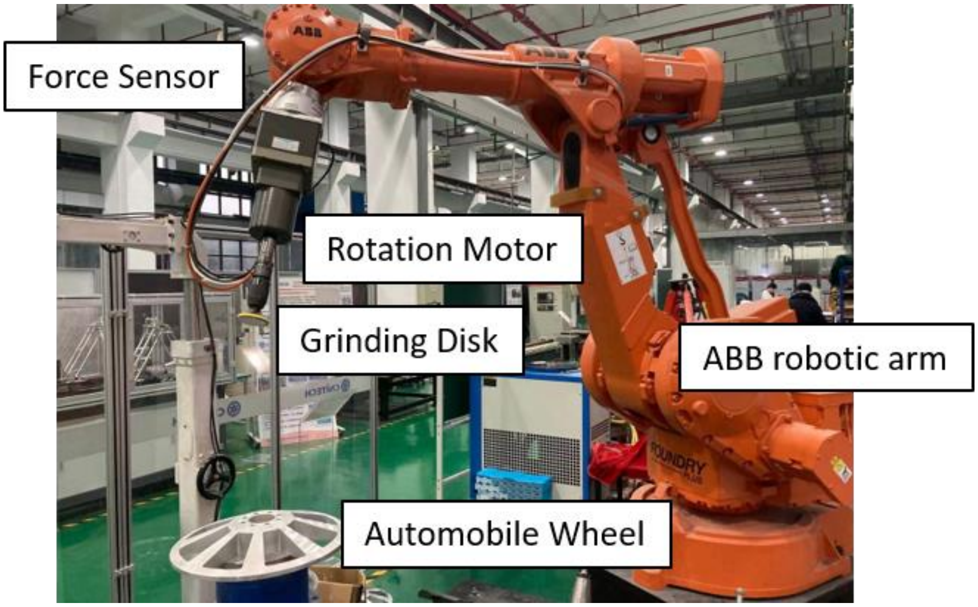


Figure 5-1 Experiment system of automobile hub grinding

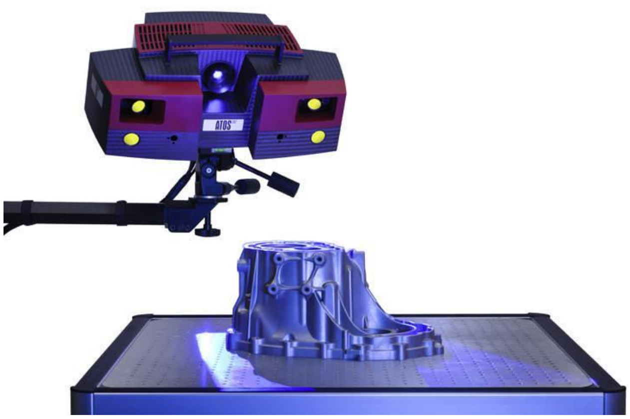


Figure 5-2 ATOS Triple Scan

First, I compare the greedy strategy and global strategy in path spacing planning. Figure 5-3 shows the result. We can tell that global strategy is better than greedy strategy. But when we use global strategy, there are too many variables to optimize. Traditional optimization methods may not work sometimes.

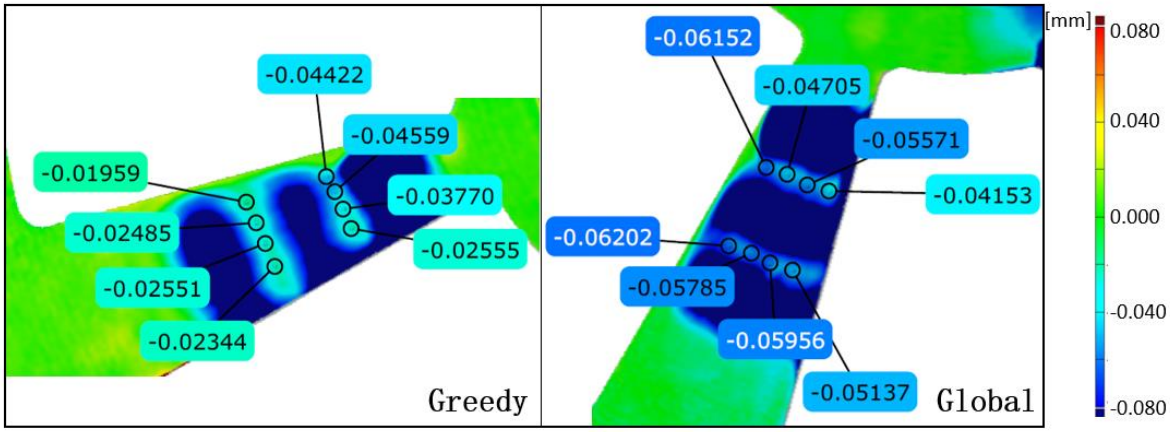


Figure 5-3 Results of experiments of greedy strategy and global strategy

Second, I utilize the proposed path planning algorithm to grind half of upper surface of the hub. The results are shown as Figure 5-4 and Figure 5-5.

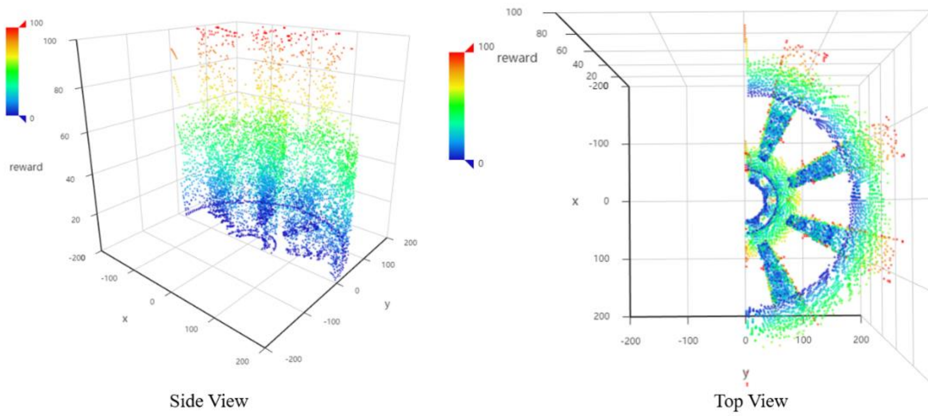


Figure 5-4 Simulation results of automobile hub grinding

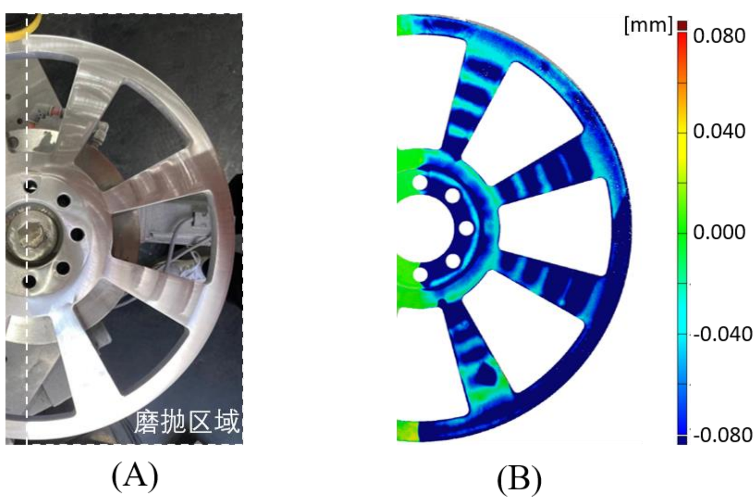


Figure 5-5 Experiment results of automobile hub grinding