Proceedings of 2019 IEEE
International Conference on Mechatronics and Automation
August 4 - 7, Tianjin, China

Path Generation for Robotic Polishing of Free-form Surfaces

Zhaosheng Li[1,2], Linlin Shang[1,2], Wei Wang[1] and Taiwen Qiu[3]
[1]*Institute of Automation, Chinese Academy of Sciences, Beijing, China*
[2]*University of Chinese Academy of Sciences*, *Beijing, China*
[3]*Shanghai Aircraft Manufacturing Co., Ltd., Shanghai, China*
{lizhaosheng2017, shanglinlin2016, wei.wang}@ia.ac.cn, qiutaiwen@comac.cc

***Abstract*- This paper develops an off-line programming system to generate tool paths for robotic polishing. The system extracts CAD data from STL format files of free-form surfaces and builds topology relations of triangular facets. We adopt direction-parallel path pattern, since it needs less degrees of freedom and is much easier to avoid the physical intervention and the dynamic singularity. We incorporate topology relations of triangular facets into the direction-parallel path generation algorithm with equal axis intervals, so this can remove the redundant data and facilitate the efficiency of path generation. Again, we propose the constant contour interval path generation algorithm, which can generate paths with equal contour intervals. The generated paths with equal contour intervals are desired for robotic polishing. We implement the off-line programming system based on Qt and OpenGL. The system can generate accurate and uniform paths of free-form surfaces, such as the turbine blade surface and the propeller blade surface.**

Index Terms - path generation; free-form surface; robotic polishing; off-line programming

I. INTRODUCTION

During the process of mold manufacturing, the surface roughness of mold affects the product quality. Molds are general polished manually, as the finishing process after milling. This process requires special technical skills and takes much time, which leads to higher cost of the molds [1]. Moreover, high-end manufacturing fields, such as turbine and propeller blades manufacturing, also requires the polishing technology. The surface quality of these work-pieces has a direct impact on the service life, stability and reliability of related equipment. To improve the surface quality, the skilled workers cost much time to polish the work-piece [2]. It is absolutely essential to develop the off-line programming system to generate tool path for robotic polishing.

There have been many researches about path generation for robotic polishing. F. Nagata et al. [3] develop a furniture polishing robot using a trajectory generator based on cutter location (CL) files. Then F. Nagata et al. [4][5][6] apply the robot to the polishing process of poly ethylene terephthalate (PET) bottle molds and get paths from the CL data generated from 3D CAD/CAM systems. K. Zhang et al. [7] utilize the six-axis machine tool to polish the blade and propose the path generation method of machining on the surface of the blade. X. Yang et al. [8] design an automatic belt grinding system to grind the blade and generate the path of grinding. Hon-yue Tam et al. [9] utilize the scanning paths method to generate the polishing path. Y. Mizugaki et al. [10] suggest a new method of Fractal path generation in a robot system of polishing metal molds. Rososhansky. M et al. [11] present a path generation method for automated polishing and apply contact mechanics for contact area modeling and analysis. Kout. A et al. [12] present a general method of offset curve construction with tool adaptive offset and an application-independent algorithmic framework of the method for work-piece surfaces represented by a triangular mesh. W.L Li et al. [13] propose a new tool path generation method for triangular meshes based on the least-squares conformal map. R. S. Freitas et al. [14] describe the methodology and steps involved in off-line programming for automatic trajectory generation. Although some tool path generation algorithms are developed, there are still plenty of challenges for the variety of free-form surfaces which have many different shapes and curvatures.

Conventional industrial robots only provide a teaching pendant as the user interface device and the teaching of free-form surface is extremely difficult and complicated. The off-line programming using the CAD models is convenient and efficient. This paper utilizes the stereolithography (STL) file which includes plenty of unstructured triangular facets to represent the CAD model. It is difficult to plan paths on the independent triangles. Therefore, we use the half edge structure to build topology relations of triangular facets. The active edge algorithm is used to extract the polishing surface. After analyzing the direction-parallel and the contour-parallel path patterns, we adopt the direction-parallel path pattern, since it needs less degrees of freedom and is much easier to avoid the physical intervention and the dynamic singularity. We incorporate topology relations of triangles into the direction-parallel path generation algorithm with equal axis intervals, so this can remove the redundant data and facilitate the efficiency of path generation. Again, we propose the constant contour interval algorithm, which can generate paths with equal contour intervals. We implement the off-line programming system based on Qt and OpenGL. The system can generate accurate and uniform paths of free-form surfaces, such as the turbine blade surface and the propeller blade surface.

II. PRELIMINARY WORK

A. Extraction of CAD Data from STL Files

978-1-7281-1699-0/19/$31.00 ©2019 IEEE

A STL file describes an unstructured triangulated surface by the unit normal and vertices (ordered by the right-hand rule) of the triangles using a three-dimensional cartesian coordinate system. According to the characteristics of the STL files, we extract CAD data from STL files using Qt program.

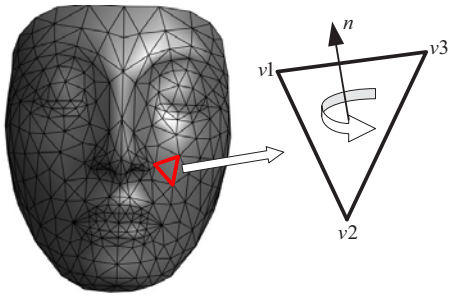


Fig. 1 The mesh surface of the face acquired from the STL file.

The data redundancy of the STL file is very large as almost every vertex is recorded repeatedly 6 times [15]. At the same time, it is difficult to analyze the free-form surface feature and generate paths on hundreds of unstructured triangular facets. It is essential to build topology relations, which can remove the redundant data and improve the efficiency of path generation.

B. Topological reconstruction

The half edge structure [16] used to build the topology relation is shown in Fig. 2.

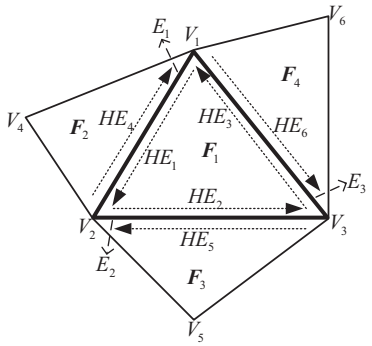


Fig. 2 The half edge structure.

An edge in the mesh surface is divided into two directed edges. Each triangle consists of three half edges and three vertices, and all of them meet the right-hand rule. Building topology relations is to create the adjacent information among triangles, half edges and vertices.

During the process of building the topology relations, the data structures of vertices, half edges and triangles need to be built. The data structures are divided into four classes: *Vertex*, *HalfEdge*, *Facet* and *STLSolid*.

The main work of the topological reconstruction process is to merge duplicate vertices, which asks for lots of searches, so the speed of searching vertices affects the speed of the topological reconstruction. The *set* container of C++ Standard Template Library is used to build topology relations, because it is very fast to search data for the set container which is implemented by binary search trees. The *set* container stores unique elements following a specific order. The value of an element is also the key used to identify it. It is essential to overload the comparison operator for the *set* container. Algorithm 1 shows the steps of topological reconstruction.

Algorithm 1: BUILD_TOPO (*verArray, heArray, fArray*)

```
1   declare vertexSet and halfEdgeSet.
2   for i in num triangles
3     for j in three vertices
4       if (j vertex in the vertexSet) continue
5       else
6         insert the vertex into the vertexSet and verArray
7         complete other relations such as the index of vertex
8         and the relation of vertex and facet
9     for k in three half edges
10      insert k half edge into the halfEdgeSet and heArray
11      if (adjacent half edge of the k in halfEdgeSet)
12        add the adjacent relation of the k half edge
13      complete other relations such as the relations of the half edge
14      and vertices, the relations of the half edge and the facet
15    insert the facet into fArray
```

C. Extraction of Polishing Surface

The process of extracting polishing surface mainly includes two steps, judging if the triangle is needed and extraction of triangles and boundary half edges.

To determine if the triangle is needed, we improve the edge-based method for defining feature boundaries which relies on the dihedral angle between two triangles [17]. The normal vector of a new triangle is n_1, the given normal vector is n_2 and the threshold is *threshold*. If

$$threshold = \frac{n_1 n_2}{|n_1||n_2|} \tag{1}$$

the new triangle is needed.

We use the active edge algorithm to extract the polishing surface. Algorithm 2 shows the steps to extract the surface.

Algorithm 2: EXTRACT_SURFACE(*exfacetSet, bdhfSet*)

```
1   declare the active half edge linked list halfedgeList.
2   find the first facet needed, add the three half edges to halfedgeList
3   and add the facet to exfacetSet.
4   while (halfedgeList is not empty)
5     search the adjacent triangle of the current half edge.
6     if(adjacent triangle dissatisfies the threshold condition)
7       add the half edge to the bdhfSet
8     else
9       insert adjacent facet into exfacetSet.
10      add the two half edges of the adjacent facet to halfedgeList
11    delete the current half edge in halfedgeList
```

III. Path Generation

A. Path Patterns

There are two main path patterns, the contour-parallel path pattern and the direction-parallel path pattern [18].

1) Contour-parallel path pattern

A series of cutting planes which are vertical to Z-axis cut the surface of the work-piece. The cutting plane intersects with the work-piece generating the color one and its boundary red line is the path of tool in Fig. 3(a).

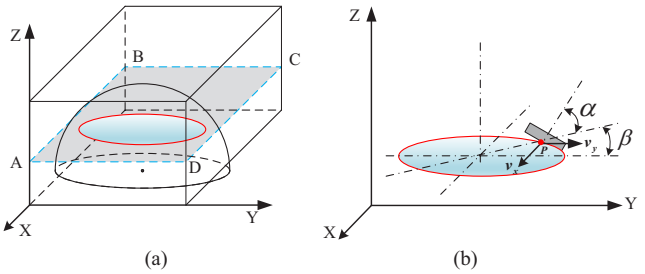


Fig. 3 The contour-parallel path pattern.

The path generation not only needs the accurate trajectory but also needs to guarantee the correct pose of the tool. The movement of the polishing tool is shown in Fig. 3(b). In the process of polishing, the feed motion of the tool includes the movement on the cutting plane and the rotation around the point *P*. The rotation motion of the tool main includes the changes of the cycloid angle and the elevation angle. The motion on the plane is that the tool moves along the X-axis and Y-axis, namely the v_x and v_y.

2) Direction-parallel path pattern

Likewise, a cluster of cutting planes which are vertical to X-axis cut the surface of the work-piece. The cutting plane intersects with the work-piece generating the color one and its boundary red line is the path of polishing tool in Fig. 4(a).

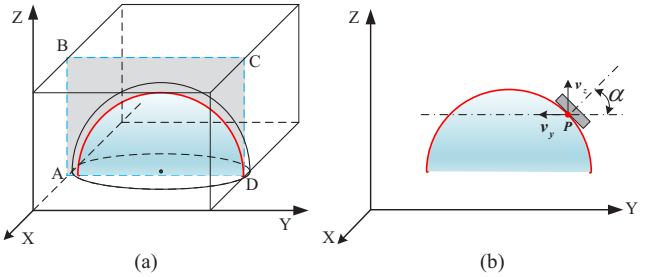


Fig. 4 The direction-parallel path pattern.

The movement of the polishing tool is shown in Fig. 4(b). The feed motion of the tool includes the movement on the cutting plane and the rotation. The rotation of the tool is the change of the α . The movement on the plane includes the motion along Y-axis and Z-axis, namely v_y and v_z.

The tool movement of the contour-parallel pattern has two linear motions and two rotations, and the tool movement of the direction-parallel pattern has two linear motions and one rotation. Compared with the contour-parallel pattern, the direction-parallel path pattern needs less degrees of freedom and is much easier to avoid the physical intervention and the dynamic singularity, so we adopt the direction-parallel pattern to generate the path.

B. Direction-parallel Paths with Equal Axis Intervals

As shown in Fig. 5, the path generation algorithm is that a series of cutting planes with equal axis intervals cut the work-piece and intersect with the work-piece generating paths.

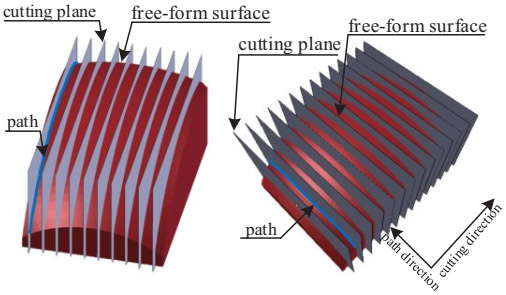


Fig. 5 Two different directions paths sliced by a series of parallel planes.

We can get boundary half edges of the free-form surface in previous sections, and this facilitate us calculate the first and the last intersections of one path. We can get the coordinate of the cutting plane after we get the range of the free-form surface. The steps of path generation are as follows.

Step 1: Calculate the *x* coordinate of the cutting plane $X[i]$. Declare the vector container *nodeVec* which stores the intersections.

Step 2: Traverse the boundary half edge array and get two half edges $bHF_i[2]$ which intersect with the cutting plane. Declare the current triangle which has the boundary half edge $bHF_i[0]$. Add the intersection between the cutting plane and the half edge $bHF_i[0]$ into the *nodeVec*.

Step 3: Traverse the three half edges of the current triangle and calculate the intersection between the plane and the half edge. If the intersection is new, add the intersection into the *nodeVec*. Else, continue.

Step 4: Update the current triangle. If the intersection last searched lies in the half edge, we update current triangle to adjacent triangle of the half edge. If the intersection lies in the vertex, we traverse the adjacent triangles of the vertex and update the current triangle to the adjacent triangle which intersects with the plane.

Step 5: Judge if the half edge last searched is the boundary half edge, namely $bHF_i[1]$. If so, end. If not, return step 3.

The steps above help us get one path, and we can get a series of paths by updating coordinates of cutting planes.

C. Constant Contour Interval Algorithm

For the direction-parallel path generation algorithm with equal axis intervals, the intervals on the contour between two paths is different as shown in Fig. 6. This algorithm only keeps axis intervals equal, which is not desired. There will be the problem of overlap or leak between two paths during the process of polishing.

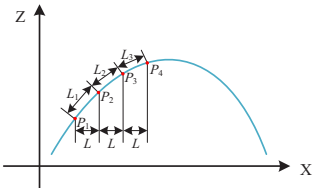


Fig. 6 The lateral view of the path on the free-form surface.

To solve the problem of the above algorithm, we propose an improved algorithm based on the contour intervals of the free-form surface. We call it constant contour interval algorithm. The steps of constant contour interval algorithm are as follows.

Step 1: Get the intersections between the cutting planes and the free-form surface. The intersections are the small black circles shown in Fig.7.

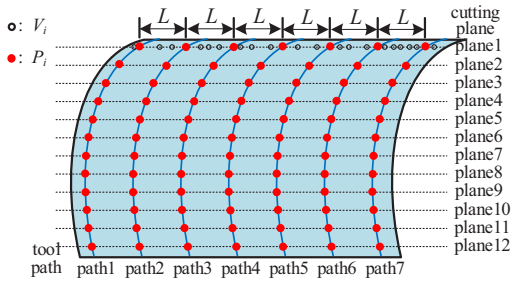


Fig. 7 The constant contour interval algorithm.

Step 2: Get the path points from each array which stores intersections between the plane and the free-form surface according to the path interval L. The path points are the red points shown in Fig. 7.

Let the first intersection be the first path point. Then iterative search the path points as shown in Algorithm 3.

Algorithm 3: SEARCH_POINTS(n, V, P, L)

```
preSum = 0, curSum = 0, P0=V0, k=1
for i = 1 to n
   preSum = curSum
   curSum = curSum + Vi-1Vi
   if (curSum < L)    continue
   else if (curSum == L)    Pk=Vi, k++, curSum=0
   else
      calculate the Pk between the Vi-1 and Vi using the
      Vi-1 Pk = L-preSum, k++, curSum=Vi-1Vi-Vi-1Pk
```

Step 3: Connect the path points obtained from the step 2 along the path direction.

Fig. 8 shows the two paths using the two methods. As is shown, constant contour interval algorithm can generate more uniform paths on the free-form surface.

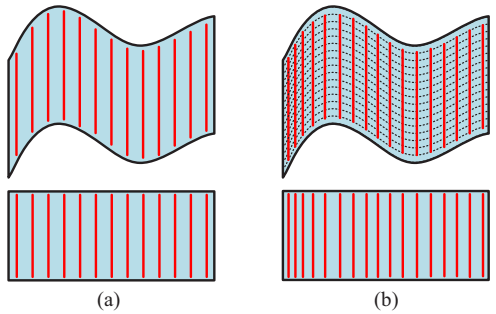


Fig. 8 The comparison of direction-parallel path generation algorithm with equal axis intervals and constant contour interval path generation algorithm. (a) The direction-parallel paths. (b) The paths with equal contour intervals.

D. Normal Vector of the Path Point

The polishing tool maintains a constant angle with the normal vector of the surface in the process of polishing. Hence, we need to calculate the normal vector of the path point. Path points have three situations, inside of the triangle, on the vertex of the triangle and on the edge of the triangle. Fig. 9 shows the last two cases.

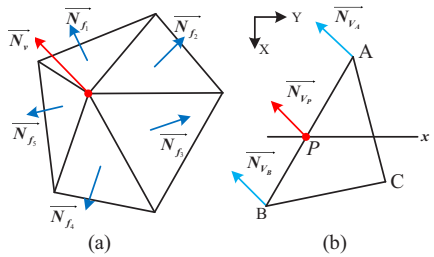


Fig. 9 The calculation of the normal vector. (a) The normal vector of the vertex. (b) The normal vector of the intersection.

The calculation methods of three cases are as follows.

- The normal vector of the path point is same with the normal vector of the triangle when the path point is inside of the triangle.
- When the path point is on the vertex of the triangle, the normal vector of the point is

$$\overrightarrow{N_v} = \frac{\sum_{i=1}^{n} \overrightarrow{N_{f_i}}}{n} \tag{2}$$

  where i is the adjacent triangle index of the vertex.
- When the path point is on the edge of the triangle, the normal vector of the point is

$$\overrightarrow{N_{V_P}} = \overrightarrow{N_{V_A}} + \frac{|\overrightarrow{AP}|}{|\overrightarrow{AB}|}\left(\overrightarrow{N_{V_B}} - \overrightarrow{N_{V_A}}\right) \tag{3}$$

IV. Algorithm Implement

A. Off-line Programming System

The off-line programming system is developed on Qt creator 4.03. In order to display the STL model and the paths generated by above algorithms, we use the OpenGL which is a cross-platform application programming interface for rendering graphics. The development of user interface software and the calculation of the path generation is on Qt 5.7.1. The path generation system is shown in Fig. 10.

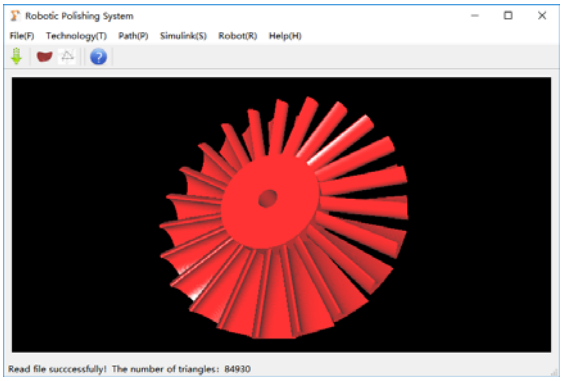


Fig. 10 The path generation system developed on Qt creator.

B. *Path Generation Results*

In order to verify the direction-parallel path generation algorithm with equal axis intervals and the constant contour interval path generation algorithm, the Robotic Polishing System generates plenty of paths of different free-form surface.

1) Direction-parallel paths with equal axis intervals

The system generates the paths of two representative surfaces, the turbine blade and the propeller blade surfaces. Fig. 11 shows the CAD model of the turbine blade and the free-form surface.

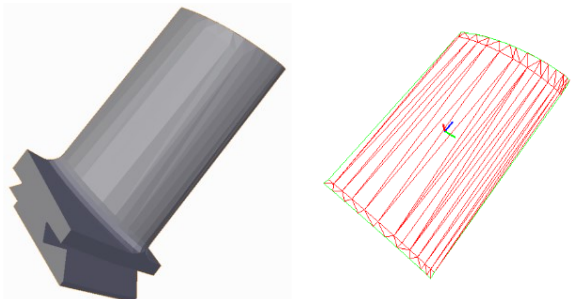


Fig. 11 The CAD model of the turbine blade and the free-form surface.

The system generates the paths of different directions and different cutting intervals in consideration of the polishing process and the polishing tool size. We get the desired paths of the turbine blade surface. The results of the path generation for the turbine blade surface are shown in Fig.12.

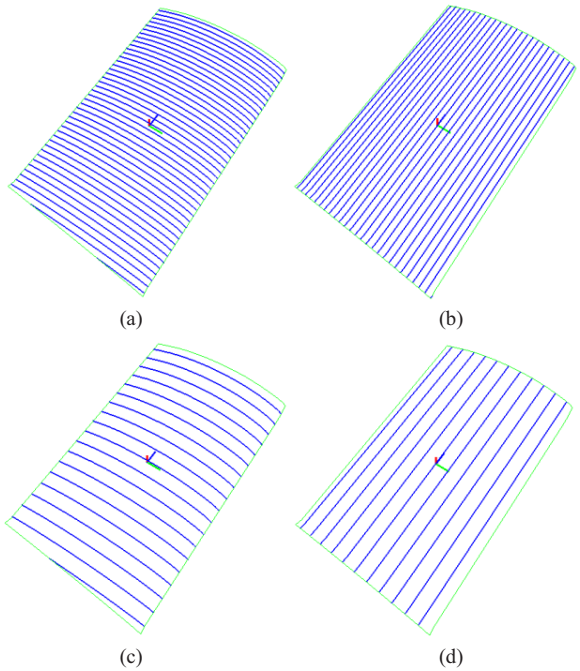


Fig. 12 The result of path generation. (a) Paths along Y-axis in a small interval. (b) Paths along Z-axis in a small interval. (c) Paths along Y-axis in a large interval. (d) Paths along Z-axis in a large interval.

The cutting planes vertical to Z-axis cut the free-form surface in a small interval shown in Fig. 12(a). The cutting planes vertical to Y-axis cut the free-form surface in a small interval shown in Fig. 12(b). The cutting planes vertical to Z-axis cut the free-form surface in a large interval shown in Fig. 12(c). The cutting planes vertical to Z-axis cut the free-form surface in a large interval shown in Fig. 12(d).

Fig. 13 shows the CAD model of the propeller blade and the free-form surface.

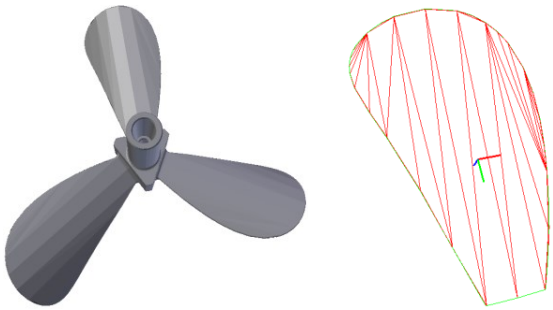


Fig. 13 The CAD model of the propeller and the free-form surface.

The system generates the paths of different directions and different cutting intervals between two paths. We get the desired paths of the propeller blade. The results of the path generation for the propeller blade are shown in Fig.14. The uniform paths cover the free-form surface of the propeller blade.

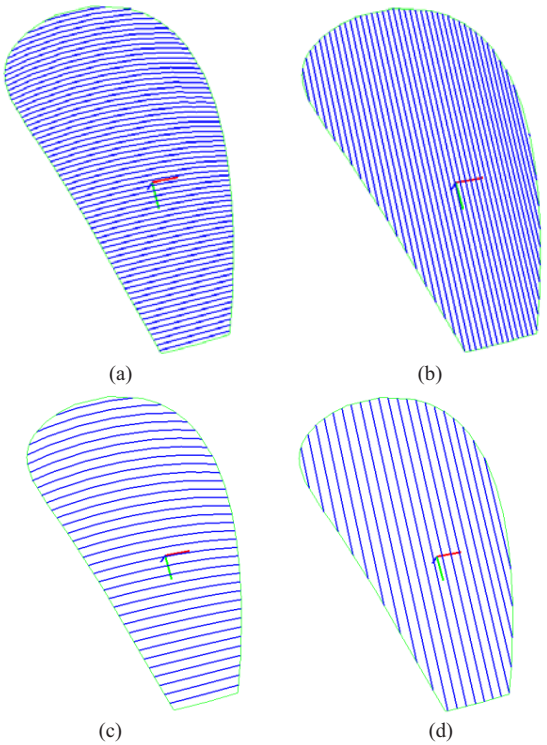


Fig. 14 The paths of the propeller blade. (a) Paths along X-axis in a small interval. (b) Paths along Y-axis in a small interval. (c) Paths along X-axis in a large interval. (d) Paths along Y-axis in a large interval.

2) Constant contour interval paths

The system can generate the constant contour interval paths using the proposed algorithm. The CAD model and the free-form surface to be polished is shown in Fig. 15.

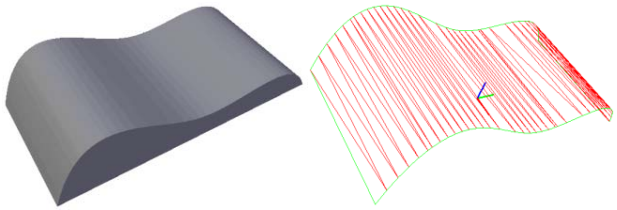
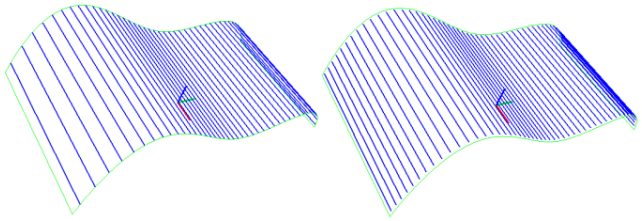


Fig. 15 The CAD model and the free-form surface.

The results of the path generation are shown in Fig. 16. The Fig. 16(a) shows the results of the direction-parallel path algorithm and Fig. 16(b) shows the result of the constant contour interval algorithm. As is shown, constant contour interval algorithm can generate more uniform paths on the free-form surface. This path generation algorithm solves the problem of overlap or leak between two paths.



(a) (b)

Fig. 16 The paths generated by two algorithms. (a) Direction-parallel path generation algorithm. (b) Constant contour interval path generation algorithm.

V. Conclusion and Future Work

In this paper, we develop an off-line programming system to generate tool paths for robotic polishing. The system extracts CAD data from STL format files of the free-form surface and builds topology relations of triangular facets. We use the direction-parallel path generation algorithm with equal intervals based on topology relations and propose the constant contour interval path generation algorithm. We implement the off-line programming system based on Qt and OpenGL. The system can generate accurate and uniform paths of free-form surface, such as the turbine blade and the propeller blade. The proposed algorithm solves the problem of overlap or leak between two paths in the process of polishing.

Future work will concentrate on improvement of path generation, polishing process, polishing system and force control of the industrial robot.

Acknowledgment

The research is sponsored by the National Commercial Aircraft Manufacturing Engineering & Technology Research Center Innovation Fund of China (COMAC-SFGS-2017-36737).

References

[1] Daqi Li, Lei Zhang, Ji Zhao, Xu Yang and Shijun Ji, "Research on polishing path generation and simulation of small mobile robot," *IEEE 2009 International Conference on Mechatronics and Automation*, pp. 4941-4945, 2009.

[2] B. Tsong-Jye Ng, Wen-Jong Lin, Xiaoqi Chen, Zhiming Gong and JingBing Zhang, "Intelligent system for turbine blade overhaul using robust profile re-construction algorithm," *ICARCV 2004 8th Control, Automation, Robotics and Vision Conference*, vol. 1, pp. 178-183, 2004.

[3] F. Nagata, K. Watanabe and K. Izumi, "Furniture polishing robot using a trajectory generator based on cutter location data," *IEEE International Conference on Robotics and Automation*, vol. 1, pp. 319-324, 2001.

[4] F. Nagata et al., "Polishing robot for PET bottle molds using a learning-based hybrid position/force controller," *2004 5th Asian Control Conference*, vol. 2, pp. 914-921, 2004.

[5] F. Nagata et al., "New finishing system for metallic molds using a hybrid motion/force control," *2003 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 2171-2175, 2003.

[6] Nagata F, Hase T, Haga Z, "CAD/CAM-based position/force controller for a mold polishing robot," *Mechatronics*, vol. 17, pp. 207-216, 2007.

[7] K. Zhang, G. Zhu, S. Liu, B. Qian, X. Zhang and C. Zhang, "Path generation for machining on surface of a blade," *2017 IEEE International Conference on Robotics and Biomimetics*, pp. 2093-2098, 2017.

[8] Xu Yang, Ji Zhao, Lei Zhang and Daqi Li, "Research on the automatic belt grinding system for machining blade with complex surface," *2010 2nd International Conference on Advanced Computer Control*, pp. 530-534, 2010.

[9] Hon-yue Tam, Osmond Chi-hang Liu, Alberet C.K. Mok, "Robotic polishing of free-from surfaces using scanning path," *Journal of Materials Processing Technology*, vol. 95, pp. 191-200, 1999.

[10] Y. Mizugaki, m. Sakamoto, T. Sata, "Fractal path generation for a metal-mold polishing robot system and its evaluation by the operability," *Ann CIRP*, vol. 42, no. 1, pp. 531-534, 1992.

[11] Rososhansky M, Xi F, "Coverage based tool-path generation for automated polishing using contact mechanics theory," *Journal of Manufacturing Systems*, pp. 144-153, 2011.

[12] Kout A, Müller, Heinrich, "Tool-adaptive offset paths on triangular mesh workpiece surfaces," *Computer-Aided Design*, 2014.

[13] WenlongLi, ZhoupingYin, YonganHuang, "Tool path generation for triangular meshes using least-squares conformal map," *International Journal of Production Research*, 2011.

[14] R. S. Freitas, E. E. M. Soares, R. R. Costa and B. B. Carvalho, "High precision trajectory planning on freeform surfaces for robotic manipulators," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3695-3700, 2017.

[15] Zhan Haisheng, Z. Ding and L. Zhou, "Connectivity Compression for meshes," *Active Media Technology - the Second International Conference*, 2014.

[16] Sieger D, Botsch M. Design, "implementation and evaluation of the surface mesh data structure," *Proceedings of the 20th International Meshing Roundtable*, pp. 533-550, 2012.

[17] Razdan A, Bae M S, "A hybrid approach to feature segmentation of triangle meshes, " *Computer Aided Design*, 2003.

[18] J. M. Zhan, X.Q. Zhou, L.Y. Hu, "Study on Path generation for Industrial Robots in Free-Form Surfaces Polishing", *Key Engineering Materials*, vols. 392-394, pp. 771-776, 2009.