ATLANTIS PRESS

Advances in Engineering Research, volume 127

3rd International Conference on Electrical, Automation and Mechanical Engineering (EAME 2018)

A Slicing Algorithm Based on Virtual Edge for 3D Printing

Yifei Hu[1,3], Jiang Xin[1,3,*], Guanying Huo[1,3], Danlei Ye[1,3], Zehong Lu[2], Bolun Wang[1,3] and Zhiming Zheng[1,3]

[1]LMIB and School of Mathematics and Systems Science, Beihang University, Beijing, China
[2]School of Mathematical Science, Peking University, Beijing, China
[3]Big Data Brain Computing Center, Beihang University, Beijing, China
[*]Corresponding author

***Abstract*—Virtual edge method for point cloud slicing in 3D printing is a very common algorithm due to the advantage of low computational complexity. However, this approach is not robust to density variations. To overcome this shortcoming, in this paper, a concept of column neighborhood is introduced to estimate whether the virtual edge model should be used in certain situations. Concretely, based on the technic of generating a set of multiple contours, we present an improved algorithm for high resolution point cloud slicing. To deal with multiple contours, a classify method based on convex hull is proposed, which reduces considerable calculations and separates each module from the others. At the last step, a method based on minimum angle is proposed to generate a single contour for each class. This algorithm is verified to be efficient by testing on kinds of typical benchmarks.**

Keywords—virtual edge; convex hull; classify

I. INTRODUCTION

Additive manufacturing (AM), also referred to as 3D printing, has attracted the interest of media, public and researchers in many fields [1]. So far, various Am processes have been developed. Although the details vary from each other, the 3D model is sliced into a set of 2.5D layered contours along the building direction in all processes [2-6]. The procedure of 3D printing mainly consists of four major steps: (1) chose the building direction, which can be constant or adaptive. (2) determine layer thickness, which results to uniform slicing or adaptive slicing. (3) generate the contour of each layer. (4) supplement support structure if necessary [7]. In the above four steps, the generation of layer contour directly determines the accuracy of the printed model.

In general, generating layered contour can be realized in three approach. In the first approach, a surface model is constructed from point cloud and then converted to a STL file format. In the second approach, a STL file format is created from point cloud directly. The third approach generate contours from point cloud without other model conversion [8]. However, there are three error sources in the first approach, namely, error between point cloud and surface, error between surface and STL file and error between STL model and layered contours. And two error sources in the second approach similarly [9]. Since there are less error source, and with the development of 3D measurement technology, accurate and dense point clouds can be easily obtained [10]. The third approach become the main research object of researchers.

As for layer contour generation based on point cloud directly, various method have been put forward. These theories can be divided into two categories: a vertical projection approach [8, 11, 12] and a virtual edge approach [13]. The difference between the two theories is the way the points are obtained on the slicing plane. The former directly projects the correlated points onto the slicing plane, and the latter constructs virtual edges according to the correlated points, and calculates the intersection point between the virtual edges and the slicing plane.

Sun et al [14] proposed virtual edge method first in 2006.the method mainly consists of three major steps: (1) Minimum distance based correlated point pair is used to generate virtual edges. (2) The intersection points between the virtual edges and the slicing plane are calculated to generate contours. (3) Edge grouping and edge junction building are proposed to link those points into layered contours. Xu et al [2] put forward a PLSP based layered contour generation, which belong to vertical projection approach. A mathematical model of PLSP of point is derived firstly, then a new weight function is given. In order to guarantee the contour generated on the slicing plane, all correlated points are projected into slicing projection along the building direction. A method based on minimum angel is proposed to generate a contour, and final contour is generated by applying the PLSP method to points of the contour. Zhang et al [15] given a contour generation based on linear fitting. All correlated points are projected onto the slicing plane first, so it belongs to vertical projection approach. A shape-error is given to control the thickness in this method, which means every time the contour is generated, the shape-error is calculated, and the thickness is adjusted if necessary. As for contour generation on slicing plane, the projected points are divided into several neighborhood groups, and the points within each group have good linearity, then a line segment is used to represent the points within a group, the contour is generated by linking all line segments based on minimum distance finally.

This paper focuses on two possible problems in the virtual edge model: (1) the number of virtual edges is between 1/3 and 1/2 of the number of correlated points, and some specific details may be ignored. (2) When the distance between the two points of a virtual edge is giant, the intersection point has a larger deviation from both two points of the virtual edge, and some errors may be introduced. However, in vertical projection method, the influence of the distance from the point to the slicing plane on the generated contour is not considered, so some errors are introduced. In this paper, two kinds of projection method are

Copyright © 2018, the Authors. Published by Atlantis Press.
This is an open access article under the CC BY-NC license (http://creativecommons.org/licenses/by-nc/4.0/).

considered, and the concept of column neighborhood is put forward. According to whether the column domain of the correlated point is empty, different projection method is selected, which can greatly reduce the error. In order to handle multiple contours, a new classify method based on convex hull is introduced, which reduces a lot of calculation and separates each module from each other. Finally a contour is generated based on minimum angel for each class.

This paper is organized as follows. In Section 2, we give the compounded projection method and the way to handle multiple contours. In Section 3, the generation of a single contour is showed. The definition of accuracy and example are given in section 4. We summarize our conclusion and give some discussion in the last section.

II. THE POINTS USED TO GENERATE CONTOURS

As mentioned earlier, in this proposed method, the points used to generate a contour or multi-contours on the slicing plane (Abbreviated as compounded projected points later), are produced by projecting the correlated points onto the slicing plane along the building direction, or intersecting the VEs with the slicing plane. Then, the compounded projected points are classified into several classes if necessary, and the compounded projected points in a class are used to generate a single contour. Section 2.1 addresses the approach to the computation of the compounded projected points, section 2.2 will show how to classify the compounded projected points into several classes.

A. Computation of the Compounded Projected Points

As shown in Figure I, P_{slice} is a slicing plane with a height of h in the point cloud, and the layer thickness is 2δ, which can be constant or adaptive. The correlated points are classified into two sets according to Z coordinates. namely, for the point $\mathrm{p}=(x,y,z)$, $\mathrm{p}\in H^{+}$, if $\mathrm{z}>h$; $\mathrm{p}\in H^{-}$, if $\mathrm{z}<h$.

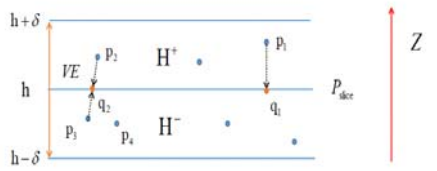

FIGURE I. THE GENERATION OF COMPOUNDED PROJECTED POINTS.

Before introducing the procedure, two definitions are defined firstly:

DEF.1 Plane distance D_{xy}: for point $\mathrm{p}=(x_1,y_1,z_1)$ and point $\mathrm{q}=(x_2,y_2,z_2)$,the plane distance between these two points is:

$$D_{\text{xy}}(p,q)=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2} \tag{1}$$

DEF.2 R-column neighborhood N_R: for a correlated point p,

$$N_R(\mathrm{p})=\{q\in H^{-}\mid D_{xy}(p,q)\le R\}\text{, if } \mathrm{p}\in H^{+} \tag{2}$$

$$N_R(\mathrm{p})=\{q\in H^{+}\mid D_{xy}(p,q)\le R\}\text{,if } \mathrm{p}\in H^{-} \tag{3}$$

The set of compounded projected points Q can be generated by the following procedures:

Step 0. Initialization: Q is empty. $P=H^{+}\ \mathrm{Y}\ H^{-}$.

Step 1. Select a point $\mathrm{p}\in P$ arbitrarily, compute $N_R(\mathrm{p})$, if $N_R(\mathrm{p})$ is empty, jump to step 2; else, jump to step 3.

Step 2. Project point p onto the slicing plane along the building direction, named the projected point as q, Q is updated as $Q\ \mathrm{Y}\{\mathrm{q}\}$,and P is updated as $P-\{\mathrm{p}\}$,jump to step 4.

Step 3. Calculate the R-column neighborhood of point p, $N_R(\mathrm{p})=\{\mathrm{p}_1,p_2,...,p_n\}$ the point p_i ,that has the minimum distance between point p ,is selected to generate a VE with point p .Named the intersection of the VE and the slicing plane as q , Q is updated as $Q\ \mathrm{Y}\{\mathrm{q}\}$,and P is updated as $P-\{\mathrm{p}\}$.

Step 4. if P is empty, jump to step 5; else, jump to step 1.

Step 5. Delete the points in Q that are repeated.

As shown in Figure I for point p_1, $N_R(\mathrm{p}_1)$ is empty, q_1 is produced by projecting point p_1 onto slicing plane along building direction. for point p_2 , $N_R(\mathrm{p}_2)=\{\mathrm{p}_3,\mathrm{p}_4\}$,as $\mathrm{d}(p_2,p_3)<d(p_2,p_4)$,point p_3 is selected to generate a VE with point p_2 ,and the intersection of the VE and the slicing plane q_2 is the compounded projected point of point p_2.

B. Handling Multiple Contours

For multiple contours, most method in other papers generate a contour firstly, then judge whether there are extra points to generate another contour. This idea requires that the algorithm of generating a single contour is accurate enough, otherwise even the number of contours generated may not be consistent with reality. The method proposed in this paper classify points into several classes firstly, and each class generates a single contour later. In this way, each module can be separated from each other, and the contours will not be inconsistent with the reality because of the instability of the single contour generation algorithm.

Before introducing the procedure, two definitions are defined firstly:

DEF.3 The distance between point and points set: the distance between point p and points set $Q=\{q_1,q_2,...,q_n\}$

is defined as:

$$\begin{aligned} &\mathrm{d}(p,Q) = d(p,q_t) \\ &d(p,q_t) \le d(p,q_i), \forall i = 1,2,...,n \end{aligned} \tag{4}$$

DEF.4 The distance between points set and points set: the distance between points set $P = \{p_1, p_2, ..., p_m\}$ and points set $Q = \{\mathrm{q}_1, q_2, ..., q_n\}$ is defined as:

$$\begin{aligned} &\mathrm{d}(P,Q) = d(p_s,q_t) \\ &d(p_s,q_t) \le d(p_j,q_i), \forall i = 1,2,...,n, \quad \forall \mathrm{j} = 1,2,...,m \end{aligned}. \tag{5}$$

The goal of this method is to classify the compounded projected points set Q into several classes satisfying the following conditions:

(1) The distance between any two classes is greater than a given threshold r.

(2) Any two points in a class can be connected by some other points in the class and some line segments whose length is less than the threshold r.

The several classes can be generated by the following procedures:

Step 0: Initialization: C_{i} are empty, $\forall \mathrm{i}$, $\mathrm{index} = 1$. Input the compounded projected points set Q.

Step 1: If Q isn't empty, jump to step 2; else, jump to step 6.

Step 2: Select a point $q \in Q$ arbitrarily, named $\{\mathrm{q}\}$ as point set D, update Q as $Q - \{\mathrm{q}\}$.

Step 3: Judge whether there is point whose distance between D is less than r in Q. If it exists, named the points set consists of these points as E, jump to step 4;else,jump to step 5.

Step 4: Update C_{index} as $C_{\mathrm{index}} \cup D$, Q as $Q - E$ .and D as E ,jump to step 3.

Step 5: Update C_{index} as $C_{\mathrm{index}} \cup D$, index as $\mathrm{index} + 1$.sort class C_{index} .jump to step 1.

Step 6: Output all $\mathrm{index}\text{-}1$ classes $C_1, C_2, ..., C_{\mathrm{index}}$.

The method stores every added points in set D in each operation, instead of being merged into C_{index} immediately, since there is no point whose distance between C_{index} is less than the threshold r in Q, the point set E can be obtained by calculating the distance between the points in Q and D only. If adds the points to C_{index} immediately, it will increase some unnecessary computation. The method proposed in this paper greatly reduces the computational complexity.

III. THE GENERATION OF A SINGLE CONTOUR

After producing the compounded projected points set Q, another method is needed to link it into several contours, since the points in Q are scattered and unordered. Since method proposed in section 2.2 has classified Q into several classes, and there is only one contour in a class, we just introduce the way to generate a single contour from a points set. For multiple contours, the same algorithm is used for each class.in this paper, a projection point band method based on angle minimum is proposed to generate a single contour.

As shown in Figure II, the input of this algorithm are points set $C = \{\mathrm{c}_1, c_2, ..., c_n\}$,find threshold K and stop threshold ε ,the output of this algorithm is points set B, whose points are ordered. a contour can be generated by linking the points in B in order.

Algorithm 1

```
1: function Contour generation(C,K,ε)
2:   //initialization
3:   B = ∅,b_0 = (0,0);
4:   i = 1;
5:   //select seed point
6:   b_i ← x_min (C);
7:   i ← i + 1;
8:   //find next point
9:   do
10:    atran(C,b_{i-1},b_{i-2});\\ coordinate conversion.
11:    N ← K_neighborhood(C,b_{i-1});\\find the K-neighborhood of point b_{i-1} in C.
12:    A ← angle_compute(N,b_{i-1});\\compute the angle paired with each point in N.
13:    b_i ← angle_min (A);
14:    i ← i + 1;
15:  while ||b_{i-1} − b_1|| > ε
16:  b_i = b_1;
17:  return B;
18:end function
```

FIGURE II. THE PSEUDO CODE OF CONTOUR GENERATION ALGORITHM

In step 2-4, initialize points set B as empty, and index $\mathrm{i} = 1$. In order to make the pseudo code coincident, named origin as b_0 , actually, b_0 does not belong to the point set B. In step 5-7, we select the first point of the contour, also known as the seed point, using function $x\_\min(C)$ returns the point that has the minimum x coordinate in the set C, making b_1 equal to this point and increasing the index one. In step 8-15, the next point is continuously searched in a cyclic manner until the distance between the current contour point and the seed point is less than the stop threshold value ε. In step 10, the coordinate of points in C have been transformed. a moving frame mounted on b_{i-1},as shown in Fig.3.and the moving frame is represented by

$$\{\mathrm{b}_{\mathrm{i}\text{-}1}, \mathrm{e}_1, \mathrm{e}_2\} \tag{6}$$

Where e_1 is vector $\overrightarrow{b_{i-1}-b_{i-2}}$, and e_2 is the external product of e_1 and building direction, namely:

$$e_2 = e_1 \times Z \tag{7}$$

In step 11, function $K\_neighborhood(C, b_{i-1})$ selects the closest K points to b_{i-1} in C, and named the set consisted of those K points as $N = \{n_1, n_2, ..., n_K\}$.In step 12, function $angle\_compute(N, b_{i-1})$ calculates the angle of each point in N. α_j is the angle between the negative axis of e_1 and the line linking b_{i-1} and n_j ,as shown in Figure 4. The formula for calculating α_j is as follows:

$$\alpha_j = \cos^{-1}\left(-\frac{n_j - b_{i-1}}{\|n_j - b_{i-1}\|} \cdot e_1\right), \text{if } n_j \in I, II \tag{8}$$

$$\alpha_j = 2\pi - \cos^{-1}\left(-\frac{n_j - b_{i-1}}{\|n_j - b_{i-1}\|} \cdot e_1\right), \text{if } n_j \in III,\ IV \tag{9}$$

In step 13-14, the point that has minimum angle is selected as the next contour point, assigning it to b_i and increasing the index one, At this point a cycle has been completed. Since the seed point has the minimum x coordinates, in theory, b_1 can be reach again. But in practical application, since the existence of the calculation error, it may not go back to the b_1. The stop condition is weakened in this paper, when the distance between the current contour point and the seed point is less than the stop threshold, the cycle can be jumped out. In order to get a closed contour, add a point $b_i = b_1$ finally. At this point, regardless of whether or not the seed point is finally reached, the final contour is the same.

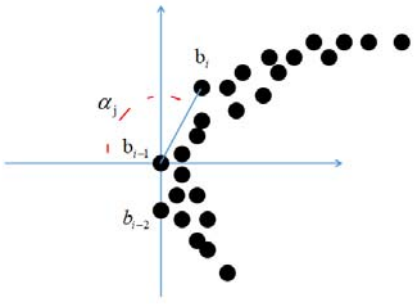

FIGURE III. THE CONTOUR EXTRACTION

As shown in Figure IV, the red points are correlated points in some layer, and the blue polygon is the contour generated by the method proposed in this paper.

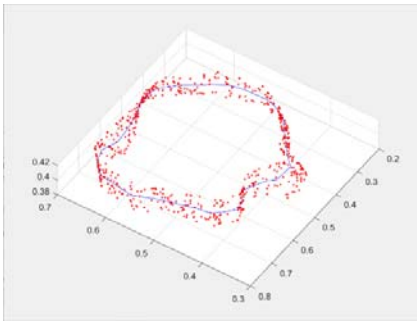

FIGURE IV. A CONTOUR GENERATED BY THIS METHOD

IV. ACCURACY OF LAYER CONTOUR

The method proposed in this paper has been coded in MATLAB language in a PC. Since there is no uniform definition for the accuracy of layer contour now.in the following, we first introduced the definition of accuracy proposed in this paper in section 4.1, then applied the method to the point cloud of a cat and analyzed the accuracy of this method in section 4.2.

A. The Definition of Accuracy

We take the points set of a layer as an example to introduce this definition. Named the points set of a layer as A, and the contour generated by the method proposed in this paper as $B = \{b_1, b_2, ..., b_n, b_1\}$,namely, the polygons connected by those points in turn. Take into account that a certain thickness along the contour of the slicing plane should be printed. As shown in Figure V, the black line segment is the distance from the point to the printed contour, and the green line segment is the distance from the point to the contour on the slicing plane. The distance of point p to the printed contour is defined as:

$$d = \min(d(p, c_n, c_1), d(p, c_i, c_{i+1}), i = 1, 2, ..., n-1) \tag{10}$$

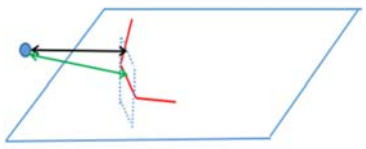

FIGURE V. THE DISTANCE FROM A POINT TO PRINTED CONTOUR

where $d(p, u, v)$ represents the distance from point p_{xy} to the line segment formed by point u_{xy} and v_{xy} , p_{xy} means the projected point of p on XY plane. It is equivalent to the minimum distance from the projected point of p on slicing plane to the contour on slicing plane. Calculate the distance d_i for each point p_i to the printed contour, the error is defined as:

$$e = \sum \frac{d_i}{n} \tag{11}$$

The smaller the value of the epsilon, the more accurate the contour generated by the method proposed in this paper.

The error of each layer can be used as an evaluation for the accuracy of each layer, and the weighted average of each layer's error can viewed as the evaluation of the accuracy of the whole point cloud.

B. *Example*

In this section, a point cloud of a cat is used as an example to test the influence of compound projection on the accuracy of layer contour generation. Here we choose the fixed layer thickness, which classified the point cloud into 40 layers, and the average error of each layer was calculated. For the convenience of comparison, the Y coordinate shown in Figure VI is the sum of the distance from each point to the print contour of each layer, namely, the cumulative error:

$$\text{sum}_i = n_i \cdot e_i \tag{12}$$

In the implementation of this example, the selection of the related paraments are:

$$\text{Layer thichness} = 0.02, \text{R} = 0.025, \text{r} = 0.07, \text{K} = 40, \varepsilon = 0.005$$

As shown in Figure VI, the red broken line is the cumulative error of each layer after using the composite projection, and the blue broken line is the cumulative error without compound projection. It can be seen that in most cases, the compounded projection reduces the accumulated error of the layer.

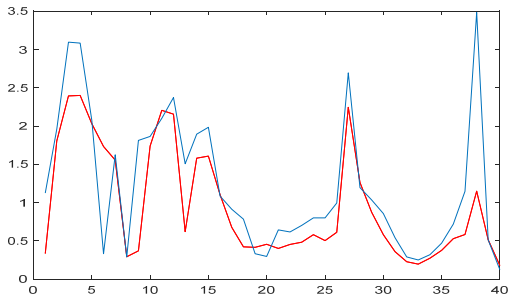

FIGURE VI. THE CUMULATIVE ERROR OF EACH LAYER

The cumulative error with the composite projection is sum=38.2369, and the cumulative error without the composite projection is sum=51.672, it decreases about 26%. The point cloud of the cat is shown in Figure VII (A), and the contour of each layer is shown in Figure VII (B).

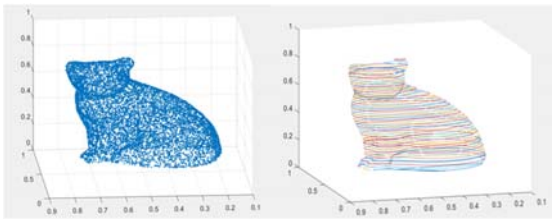

FIGURE VII. (A) THE POINT CLOUD OF A CAT. (B) THE CONTOUR OF EACH LAYER

V. CONCLUSION

This paper proposes an improved PPB method to generate contour of each layer from a point cloud for 3D printing. The proposed method generate contours from points directly, which skips the complicated model conversion, reduce the time to print the products, and avoid the error induced by model conversion. In order to avoid the two possible problems in traditional virtual edge method, compounded projection is proposed in this paper. As for handling multiple contours, a new classify method based on convex hull is introduced, which separates each module from each other. This method resulted in generating contours more ideally compared to PPB method without compounded projection. This has been demonstrated by our experiment in section 4, the error of the whole point cloud is decreased about 26%. In addition, the classify method has the ability to classify points set into several classes, which can been seen in section 4.

ACKNOWLEDGMENT

This work is supported by the Chinese National Science Foundation under Grant No.11290141.

REFERENCES

[1] W. Gao, Y. Zhang, D. Ramanujan, The status, challenges, and future of additive manufacturing in engineering, Comput. Aided Des.69 (2015) 65-89.

[2] Jingting Xu, Wenbin Hou, Yuwen Sun, Yuan-Shin Lee, PLSP based layered contour generation from point cloud for additive manufacturing, Robotics and Computer-Integrated Manufacturing, 49 (2018) 1-12.

[3] Tianyun Yuan, Xiaobo Peng, Dongdong Zhang, Direct rapid prototyping from point cloud data without surface reconstruction, Computer-Aided Design and Application, 2018, VOL.15, NO.3, 390-398.

[4] G. H. Liu, Y. S. Wong, Y. F. Zhang, Error-based segmentation of cloud data for direct prototyping, Comput. Aided Des. 35 (7) (2003) 274-288.

[5] V. K. Kumbhar, P. M. Pandey, P. V. M. Rao, Improved intermediate point curve model for integrating reverse engineering and rapid prototyping, Int. J. Adv. Manuf. Technol. 37 (5-6) (2008) 553-562.

[6] William Oropallo, Les A. Piegl, Paul Rosen, Khairan Rajab, Point cloud slicing for 3D printing, Computer-Aided Design and Application, 2018, VOL.15, NO.1, 90-97.

[7] Rodrigo Minetto, Neri Volpato, Jorge Stolfi, Rodrigo M. M. H. Gregori, Murilo V. G. da Silva, An optimal algorithm for 3D triangle mesh slicing, Computer-Aided Design, 92 (2017) 1-10.

[8] LEE. K. H, WOO. H, Direct integration of reverse engineering and rapid prototyping. Computers & industrial engineering. 38(1):21-38.

[9] Y. F. Wu, Y. S. Wong, H. T. Loh and Y. F. Zhang, Modeling cloud data using an adaptive slicing approach, Computer Aided Design, 2004, Vol. 36, No. 3,.231-240.

[10] J. Sladek, P. M. Blaszczyk, M. Kupiec, The hybrid contact-optical coordinate measuring system, Measurement 44 (3) (2011) 503-510.

[11] G. H. Liu, Y. F. Wong, H. T. Loh, Modeling cloud data for prototype manufacturing, Material Processing Technology, 2003, Vol. 138, No. 1-3, 53-57.

[12] H. Y. Shin, S. Y. Park, E. J. Park, Direct Slicing of a point set model for Rapid Prototyping, Proceeding of CAD04, 2004.

[13] Jingting XU, Wenbin Hou, Hongzhe Zhang, An improved virtual edge approach to slicing of point cloud for additive manufacturing, Computer-Aided Design and Application, 2018, VOL. 15, NO. 3, 330-336.

[14] Y. W. Sun, D. M. Guo, Z. Y. Jia, B-spline surface reconstruction and direct slicing from point clouds, Int. J. Adv. Manuf. Technol. 27 (9-10) (2006) 918-924.

[15] Y. F. Zhang, Y. S. Wong, H. T. Loh, Y. F. Wu, An adaptive slicing approach to modelling cloud data, Journal of Materials Processing Technology 140 (2003) 105-109.