Title:
An improved Virtual Edge Approach to Slicing of Point Cloud for Additive Manufacturing

Authors:
Jinting Xu, xujt@dlut.edu.cn, Dalian University of Technology
Wenbin Hou, houwb@dlut.edu.cn, Dalian University of Technology
Hongzhe Zhang, zhanghongzhe@dlut.edu.cn, Dalian University of Technology

Keywords:
Additive manufacturing, Point cloud, Slicing, Virtual edge

DOI: 10.14733/cadconfP.2017.212-216

Introduction:
Additive manufacturing (AM) is an innovative technology to fabricate the parts with complex shapes, lately has been becoming the focus of media and gained the increasing attentions of the researchers in various fields [2]. Unlike the traditional manufacturing processes, such as milling which removes the unwanted materials from a blank, AM builds up a part through the deposition of the materials layer by layer. This distinctive feature of AM brings many advantages, such as the abilities to manufacture the parts with complex internal structure, and it can fabricate cost-effectively the components made of the expensive materials, such as titanium and nickel, etc., in aerospace industry, because, for such parts, the traditional milling usually suffers from an extremely high material removal ratio [1]. So far, various AM processes have been developed, including stereolithography apparatus (SLA), fused deposition modeling (FDM), selective laser sintering or melting (SLS/SLM), electron beam melting (EBM), etc. Although the mechanical details of these processes are different from each other, they all require slicing 3D model of the part into a set of 2.5D layered contours along the building direction [8]. At present, the used model widely in AM is parametric model, such as B-spline or Bézier surface, and STL model, but they both require the model conversion from the point cloud. Although many works have been done for the solution of the parametric or STL model construction, a completely automated and satisfactory solution is still challenging and far from being fully automatic even for the high-quality data [3], [11].

Recently, with the rapid development of 3D measuring technology, dense and accurate point cloud, which can represent more exactly the geometry of a part than before, has been nicely available. Under this situation, slicing directly the point cloud is considered to be a promising alterative since it bypasses completely the conversion processes of the parametric and STL models from the point cloud, thus avoiding the accuracy loss induced by the model conversion. However, compared with the parametric model and STL model, the point cloud only provides the position information of the discrete points of the nominal model, so that slicing directly the point cloud is not easy task. In the past decade, some valuable works, which devote to slice directly the point cloud, have been proposed, and these methods can be classified onto three categories: projection point band (PPB) based method [4], [5], [10], point projection (PP) based method [7], [11] and virtual edge (VE) based method [6], [9]. PPB based method subdivides the point cloud into a number of the layers with a thickness perpendicular to the building direction and then the data points between two neighboring layers are projected onto a plane, forming PPB. Subsequently, the data points in PPB are sorted and compressed by keeping the featured points. Finally, the featured points are linked into a polygon contour [4], [10]. This method is straightforward and easy to be realized, but it can generally result in a trade-off between the projection error and the truncation error. In PP based method [7], [11], moving least square (MLS) surface is used to represent the nominal surface of the point cloud, and the intersection point of MLS surface and a specified line is computed as the contour point. This method has good computing accuracy, but it involves complex nonlinear optimization. Compared with the above methods, VE based method does not involve complicated calculation, but uses a set of the minimum distance-based correlated point pair (MDCPP) to construct the VEs, and then the contour points are obtained by slicing the VEs using the slicing plane. This method is straightforward mathematically and very easy to be implemented, but in Sun's method [9], a point can be not shared by multiple VEs; this may result in the potential accuracy loss of the contour, especially when processing the non-uniformly distributed data points. In [6], to some extent, Park et al improved VE based method using the density gauge sphere and the point supplementing technique, but excessive number of VEs may cause vibration of the slicing contour, so that the obtained contour has to be filtered and smoothed. In order to solve these problems and make VE based method be able to process well the non-uniformly distributed data points that are countered in practical engineering, an improved VE approach is proposed in this paper to directly slice the point cloud for generating the slicing contour for AM.

Main Idea

In this proposed method, the initial contour point are produced by intersecting the VEs and the slicing plane, the scattered initial contour points are then sorted and divided into multiple contours if there are multiple contours. Subsequently, the positions of initial contour points are adjusted onto the nominal surface and additional points are inserted between the sparse contour points, thereby forming the final slicing contours.

Initial contour point determination

As shown in Fig. 1, $\mathrm{P}^{(i)}$ is the ith slicing plane with height h and the point set between the planes $\mathrm{P}_{+}^{(i)}(h+\Delta\delta)$ and the planes $\mathrm{P}_{-}^{(i)}(h+\Delta\delta)$ is its correlated point set $\mathrm{C_P}$ which consists of two sets of point, i.e., Q_{+} between $\mathrm{P}_{+}^{(i)}$ and $\mathrm{P}^{(i)}$, and Q_{-} between $\mathrm{P}_{-}^{(i)}$ and $\mathrm{P}^{(i)}$. The MDCPPs are identified between Q_{+} and Q_{-} by the following procedures. For arbitrary one point $\boldsymbol{q}^{(r)}$ in Q_{+}, its closest point in Q_{-} is first determined and are assumed be $\boldsymbol{q}^{(t)}$, as shown in Fig. 1, then the closest point of $\boldsymbol{q}^{(t)}$ in Q_{+} is searched and are assumed be $\boldsymbol{q}^{(s)}$. If $\boldsymbol{q}^{(s)}$ coincides with $\boldsymbol{q}^{(r)}$, $\boldsymbol{q}^{(r)}$ and $\boldsymbol{q}^{(t)}$ constitutes MDCPP, otherwise $\boldsymbol{q}^{(s)}$ and $\boldsymbol{q}^{(t)}$ constitutes MDCPP. Assume that $\left(\boldsymbol{q}^{(s)},\boldsymbol{q}^{(t)}\right)$ is one MDCPP, the intersection, i.e. initial contour point, of the connecting line of $\boldsymbol{q}^{(s)}$ and $\boldsymbol{q}^{(t)}$ and the slicing plane $\mathrm{P}^{(i)}$ can be calculated by the following equation,

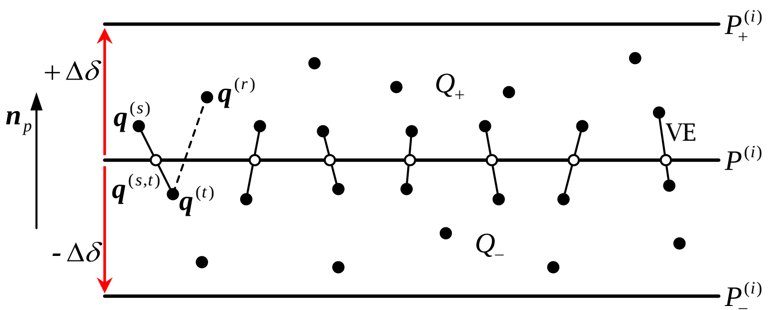

Fig. 1: Initial contour point determination based on VE

$$\boldsymbol{q}^{(s,t)} = \boldsymbol{q}^{(t)} + \frac{\left(\boldsymbol{p}_0 - \boldsymbol{q}^{(t)}\right)\cdot \boldsymbol{n}_p}{\left(\boldsymbol{q}^{(s)} - \boldsymbol{q}^{(t)}\right)\cdot \boldsymbol{n}_p}\left(\boldsymbol{q}^{(s)} - \boldsymbol{q}^{(t)}\right) \tag{1}$$

where $\boldsymbol{p}_0$ is one point on the slicing plane and $\boldsymbol{n}_p$ is the unit normal vector of the slicing plane. These above constructing MDCPP and calculating the initial contour point are performed repeatedly until all point in Q_{+} and Q_{-} are processed. Since the VEs are randomly constructed, the obtained initial contour points are scattered and non-uniformly distributed.

Initial contour point sortation

After extracting the scattered initial contour points, they need to be connected into a closed contour. If there are two or more contours, these contours also need to be recognized correctly. In this paper, a tracing algorithm based on angle detection is proposed to sort the initial contour points and recognize multiple contours.

Assume that the initial contour points set is P_C, the point in P_C, that has the minimum x coordinate, is selected as the starting point of one contour to be connected, which is denoted by $\boldsymbol{p}_{C,0}$. Then, a moving frame, on the slicing plane, is mounted on $\boldsymbol{p}_{C,0}$ and this frame is modeled by

$$\left\{\boldsymbol{p}_{C,0}; \boldsymbol{e}_1^{(M)}, \boldsymbol{e}_2^{(M)}\right\} \tag{2}$$

where the coordinate axes $\boldsymbol{e}_1^{(M)}$ and $\boldsymbol{e}_2^{(M)}$ are consistent respectively with X and Y coordinate axes of the model coordinate system. Then, the next contour point is identified if the angle φ_i is minimum between the negative axis of $\boldsymbol{e}_1^{(M)}$ and the line that connects $\boldsymbol{p}_i$ and $\boldsymbol{p}_{C,0}$, where $\boldsymbol{p}_i$ is one point in K-neighbor of $\boldsymbol{p}_{C,0}$ and φ_i is calculated by the following equation,

$$\varphi_i = \begin{cases} \cos^{-1}\left(-\dfrac{\boldsymbol{p}_i - \boldsymbol{p}_{C,j-1}}{\left\|\boldsymbol{p}_i - \boldsymbol{p}_{C,j-1}\right\|} \cdot \boldsymbol{e}_1^{(M)}\right), & \boldsymbol{p}_i \in \text{I, II quadrants} \\ 2\pi - \cos^{-1}\left(-\dfrac{\boldsymbol{p}_i - \boldsymbol{p}_{C,j-1}}{\left\|\boldsymbol{p}_i - \boldsymbol{p}_{C,j-1}\right\|} \cdot \boldsymbol{e}_1^{(M)}\right), & \boldsymbol{p}_i \in \text{III, IV quadrants} \end{cases} \tag{3}$$

where $\boldsymbol{p}_{C,j-1}$ stands for the jth contour point. Once the contour point $\boldsymbol{p}_{C,j}$ is obtained, it is used as the new starting point for finding the next contour point. The direction vector of the line linking $\boldsymbol{p}_{C,j-1}$ and $\boldsymbol{p}_{C,j}$ is used as the $\boldsymbol{e}_1^{(M)}$axis of the jth moving frame mounted on $\boldsymbol{p}_{C,j}$. Then, the procedures of searching contour points are performed repeatedly until the starting point $\boldsymbol{p}_{C,0}$ is reached again. After a closed contour is gained by the above algorithm, these contour points, as well as the points, which are inside this contour, are all removed from the initial contour point set P_C. If P_C is not empty, those remaining points will be as a new initial contour point set to initiate the procedure of sorting the contour point. The above searching and judging procedures are iterated repeatedly until all points in are processed. As a result, the multi-contours are automatically recognized.

Contour point supplement and adjusting

Actually, the point cloud density is non-uniformly distributed, it possibly causes that there are no contour points at some area where the correlated point set of the slicing plane is only on its one side, as shown in Fig. 2. Moreover, since the contour points are only the intersections of the VEs and the slicing plane, it is difficult to ensure that the contour points lie accurately on the nominal surface. Generally, the lower the point cloud density, the lager the tolerance between the contour points and the nominal surface, thus the position of the initial contour point needs to be adjusted so that it can lies on the nominal surface as accurately as possible. In addition, the gap between two consecutive contour points, as shown in Fig. 2 also requires supplying additional contour points for achieving satisfactory contour tolerance.

The gap is identified if the distance between two consecutive points exceeds the specified distance d_s by users, then a line segment is added to link the two contour points, as shown in Fig. 2, and some points are interpolated on this line segment with the interval of d_s, thus forming relatively uniformly distributed contour points sequence. After the supplement contour points are added, the initial conto-

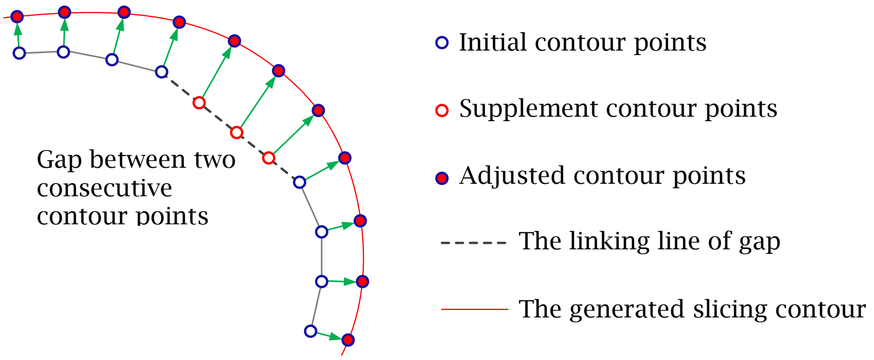

Fig. 2: Initial contour point adjusting and supplement

ur points can be adjusted onto the nominal surface using the following method. Assume that $\boldsymbol{q}_j, j = 0,1,\cdots,k$ is the K-neighbor points of the initial contour point $\boldsymbol{p}_i$, then a simple quadric surface is used to approximate the local nominal surface of K-neighbor points. First, a local coordinate system, $\xi^{(L)} = \left\{\boldsymbol{p}_i; \boldsymbol{e}_1^{(L)}, \boldsymbol{e}_2^{(L)}, \boldsymbol{e}_3^{(L)}\right\}$, is mounted on $\boldsymbol{p}_i$, where $\boldsymbol{e}_3^{(L)}$ is the unit normal vector of $\boldsymbol{p}_i$ which can easily obtained using least squares fitting method, $\boldsymbol{e}_1^{(L)}$ is the unit vector of the projection of the line linking $\boldsymbol{p}_i$ and one arbitrary point in its K-neighbor on the tangent plane of $\boldsymbol{p}_i$ and $\boldsymbol{e}_2^{(L)}$ is the cross product of $\boldsymbol{e}_3^{(L)}$ and $\boldsymbol{e}_1^{(L)}$. K-neighbor points $\boldsymbol{q}_j, j = 0,1,\cdots,k$ is transformed into $\boldsymbol{q}_j^{(L)}, j = 0,1,\cdots,k$ in $\xi^{(L)}$. Assume that the quadric surface is $z^{(L)} = z\left(x^{(L)}, y^{(L)}\right)$, then the least squares objective function of fitting the quadric surface of K-neighbor points can be written as

$$E = \min\left[\sum_{j=1}^{k} w_j \left\| z^{(L)}\left(x_j^{(L)}, y_j^{(L)}\right) - z_j^{(L)} \right\|^2\right] \tag{4}$$

where w_j is weight of $\boldsymbol{q}_j^{(L)}$. Generally, the closer $\boldsymbol{q}_j^{(L)}$ from $\boldsymbol{p}_i$, the larger its weight, thus the following weight function is adopted

$$w_j = \exp\left[-d_j \Big/ \left(\frac{1}{k}\sum_{r=1}^{k} d_r\right)\right] \tag{5}$$

In Eq. (5), d denotes the distance from one point in K-neighbor point set to the initial contour point $\boldsymbol{p}_i$. Once the quadric surface is fitted, the initial contour point can be adjusted onto the nominal surface using the following equation

$$\boldsymbol{p}_{i,\text{contour}} = \boldsymbol{p}_i + z^{(L)}(0,0)\boldsymbol{e}_3^{(L)} \tag{6}$$

Examples

The algorithms proposed in this paper have been coded in C++ language and implemented in a PC. To display clearly the layered contours, a big slicing thickness is adopted when slicing the point cloud. Fig. 3 shows three examples of slicing the complex point clouds using the proposed method. It is seen that our method can produce nicely the layered contour, not requiring any model conversion from the point cloud, and the problem of the multi-contours on complex model can be also processed automatically.

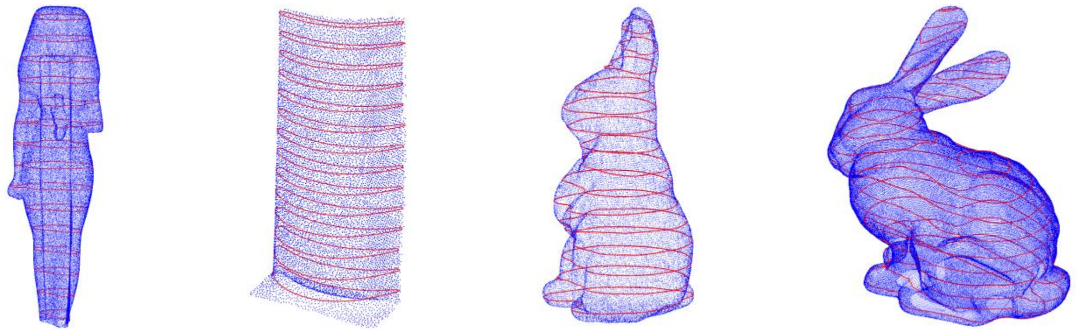

Fig. 3: Examples of slicing the point cloud

<u>Conclusions</u>

An improved VE method is proposed. In this method, the gap problem that results from non-uniformly distributed density of point cloud between two consecutive initial contour points is solved by supplying additional contour points, and the initial contour point is also adjusted onto the nominal surface of the point cloud. In addition the sorting algorithm of the initial contour point not only guarantees that the scattered initial contour point can be linked into a closed contour but also makes the proposed method have the ability of recognizing and extracting the multiple contours on complex models. These are all confirmed by the given examples.

Acknowledgements
This work is supported by the National Natural Science Foundation of China (Grant No. 51575086).

References:

[1] Ding, D.; Pan, Z.; Cuiuri, D.; Li, H.; Larkin, N.; Duin, S.: Automatic multi-direction slicing algorithms for wire based additive manufacturing, Robotics and Computer-Integrated Manufacturing, 37, 2016, 139-150. http://dx.doi.org/10.1016/j.rcim.2015.09.002

[2] Gao, W,; Zhang, Y,; Ramanujan D.; Ramania, K.; Chen, Y.; Williams, C. B.; Wang, C.; Shin, Y.; Zhang, S.; Zavattieri P. D.: The status, challenges, and future of additive manufacturing in engineering, Computer-Aided Design, 69, 2015, 65-89. http://dx.doi.org/10.1016/j.cad.2015.04.001

[3] Li, Y.; Ni, J.: Constraints based nonrigid registration for 2D blade profile reconstruction in reverse engineering, ASME, Journal of Computing and Information Science in Engineering, 9(3), 2009, 031005-031005-9. http://dx.doi.org/10.1115/1.3184602

[4] Liu, G. H.; Wong, Y. S.; Zhang, Y. F.; Loh, T. H.: Modelling cloud data for prototype manufacturing, Journal of Materials Processing Technology, 138(1), 2003, 53-57. http://dx.doi.org/10.1016/S0924-0136(03)00048-7

[5] Javidrad, F.; Pourmoayed, A. R.: Contour curve reconstruction from cloud data for rapid prototyping, Robotics and Computer-Integrated Manufacturing, 27(2), 2011, 397-404. http://dx.doi.org/10.1016/j.rcim.2010.08.008

[6] Park, H. T.; Chang, M. H.; Park, S. C.: A slicing algorithm of point cloud for rapid prototyping, Proceedings of the 2007 Summer Computer Simulation Conference, San Diego, California, USA, July, 2007, pp. 24.

[7] Qiu, Y.; Zhou, X.; Qian, X.: Direct slicing of cloud data with guaranteed topology for rapid prototyping, International Journal of Advanced Manufacturing Technology, 53(1-4), 2011, 255-265. http://dx.doi.org/10.1007/s00170-010-2829-6

[8] Steuben, J. C.; Iliopoulos, A. P.; Michopoulos, J. G.: Implicit slicing for functionally tailored additive manufacturing, Computer-Aided Design, 77, 2016, 107-119. http://dx.doi.org/10.1016/j.cad.2016.04.003

[9] Sun, Y. W.; Guo, D. M.; Jia, Z. Y.; Liu W. J.: B-spline surface reconstruction and direct slicing from point clouds, International Journal of Advanced Manufacturing Technology, 27(9-10), 2006, 918-924. http://dx.doi.org/10.1007/s00170-004-2281-6

[10] Wu, Y. F.; Wong, Y. S.; Loh, H. T.; Zhang, Y. F.: Modelling cloud data using an adaptive slicing approach, Computer-Aided Design, 36(3), 2004, 231-240. http://dx.doi.org/10.1016/S0010-4485(03)00097-6

[11] Yang, P.; Qian, X.: Adaptive slicing of moving least squares surfaces: toward direct manufacturing of point set surfaces. ASME, Journal of Computing and Information Science in Engineering, 8(3), 2009, 031003-031003-11. http://dx.doi.org/10.1115/1.2955481