Research on fast grouping slice algorithm for STL model in rapid prototyping

Renqiang Tian¹, Shaogang Liu and Yifei Zhang

School of mechanical engineering, Tianjin University of Science and Technology, 300222, Tianjin, China

¹ E-mail: trq163yx@163.com

Abstract. In this paper, a new algorithm for direct slicing of triangular facet grouping slices based on STL model in rapid prototyping is proposed. A triangular facet matrix is created by reading the binary format information of the STL model. Firstly, the binary search algorithm is used to search the triangular facet intersecting with the slice plane, and the grouping matrix is established. Then according to the slice plane intersecting with the triangular plane, the intersection line segment is obtained. Finally, the depth-first search algorithm is used to optimize the contour lines with different heights. The algorithm reduces the judgment of the position relationship between the slice plane and the triangular facet and avoids the establishment of the topological information relationship between the triangular facets. It has obvious advantage in slice time.

1. Introduction

Rapid prototyping technology is also called Additive Manufacturing, the essence of which is to add raw materials layer by layer to form solid model. In the rapid prototyping technology, the STL file format has become a standard file interface. During printing, the three-dimensional CAD model must be converted to the STL file format. The triangular facets of the STL model and the slice plane are intersected to generate contour lines, and each slice contour line is used to represent the print path of the section of the part [1]. Then, according to the path of the contour line, the raw materials are superposed sequentially from bottom to top, and the processing is repeated until the complete solid part is printed.

The slicing process is an important part of the rapid prototyping process step. At present, the slice algorithm used in rapid prototyping control software is mostly based on the slicing algorithm of STL model [2]. In this method, most triangular facets may not intersect with the slice plane. When calculating the contour lines of each slice plane, it is necessary to traverse all the triangular facet grids, which results in large data processing of the contour lines and reduces the searching efficiency. Following, there is adjacency relationship between the triangular facets, that is, the vertices and edges will have duplicate redundant data. Calculating the intersection of the triangular facet and the slice plane requires twice intersecting operations [3]. This result in a large amount of data processing, increases the intersection operation, and consumes a lot of memory space.

In order to improve the slicing efficiency, it is necessary to preprocess the triangular facets in the STL model before slicing. At present, there are two most widely used preprocessing methods: One is the STL slice algorithm based on the geometric feature of the triangular facet. The other is based on STL geometric model topology information slicing algorithm [4].
In the literature [2], [4] and [5], although the judgment of the relationship between the triangular facet and the slice plane is reduced and the search speed of the triangular facet intersecting with the slice plane is increased. The sorting of the triangular facets and the processing of the topological information between adjacent triangular facets are very time-consuming tasks. Especially, in the STL slicing algorithm based on the geometric features of triangular facets. Because the boundary of classification is not clear, there will be an invalid position relation between triangulation and tangent plane, which leading to the slicing time increases, reducing the slicing efficiency.

Considering the above characteristics, in this paper, a new slicing algorithm is proposed, that is, given a slice plane, search for intersecting triangular facets. In this algorithm, all the triangular facets that intersect with the current slice plane need to be searched, while the non-intersecting triangles do not need to search [6]. In order to reduce the slicing time, this paper proposes a grouping algorithm for triangular facet with intersecting planes before slicing.

In order to realize the algorithm, the 3D CAD model must be transformed into the most widely used STL file format in rapid prototyping. STL files mainly have two formats of ASCII and binary, their data structure is simple, the layered processing is convenient, and the readability is strong [1]. This paper uses small storage space, more compact, readable strong for the binary format.

2. The realization process of slicing algorithm in this paper
Firstly, the binary information of the STL model is read, the three vertexes coordinates of each triangular facet and a normal vector stored in the matrix. In the slicing direction, the z coordinates of the lowest and the highest vertices of each triangular facet are defined as $z_{\text{min}}$ and $z_{\text{max}}$, respectively, and a new matrix is established. Secondly the triangular facets are grouped according to the new matrix, and the triangular facets intersected with the current slice plane are placed in the same group. Then the intersection of triangular facet in the same group is calculated and the intersecting line segment is obtained. Finally, remove the redundant information between the lines segment, the contours are obtained by connecting the intersecting lines of the slice plane. The specific flow chart of this algorithm is shown in figure 1.

![Flow of slicing algorithm in this paper](image_url)
3. Triangular facet grouping algorithm

3.1. The idea of grouping algorithm

Assuming the STL model is divided into n layers of slice plane, will be the maximum number of STL triangular facet in the model into n groups. Place all triangular facets intersecting with the current slice plane in the same group. According to the slice plane to search the intersecting triangular facets, a dynamic array matrix is used to store the current intersecting triangular facets. The row vector in the matrix is planes, and the column vector is all the triangular facets intersected with each slice plane, such groups are called grouping matrices [6].

3.2. The specific steps for grouping algorithm

On the basis of the slicing algorithm of STL model, the binary search algorithm is proposed to realize the fast grouping of triangular facets. The basic idea is to constantly split the array by half, and compare the middle element of array with the target each time. If there is equivalent target in the array, it returns true, otherwise it returns false and does not return to the lookup location. Repeat this method, and half of the data can be eliminated each time until all the slice planes matching are completed [7].

The flow chart of the grouping algorithm is shown in figure 2. Specific steps are as follows:

1. Read the STL file, create the triangular facet matrix, create new triangular facet, define the slice thickness.
2. Create a node matrix with \( z_{\text{min}} \) in a new triangular facet.
3. Set the boundary conditions \( \text{low} = 1, \text{high} = \text{size(slices,2)} \).
4. Searching for the matching of \( z_{\text{min}} \) and \( \text{slice(mid)} \) of triangular facet by binary search algorithm.
5. Check high with binary search, extract the new triangular facet \( z_{\text{max}} \) and update the node matrix, set the boundary conditions \( \text{low} = 1, \text{high} = \text{size(slices,2)} \).
6. Searching for the matching between \( z_{\text{max}} \) and \( \text{slice(mid)} \) of triangular facet by binary search algorithm.
7. Change the search interval \([\text{low}, \text{mid}-1]\) or \([\text{mid}+1, \text{high}]\).
8. The slicing plane intersects with the triangular facet is stored in the \( z_{\text{triangles grouping matrix}} \).
9. Exit to find.

**Figure 2.** Find the slice plane and triangular facets intersect flow chart.

Step 1, firstly, the STL grid information is read, and the three vertex coordinates and normal vectors of each triangular facet are stored in the matrix, where each row vector is a slice plane. Then the \( \text{min}_z \) and \( \text{max}_z \) of each triangular facet are stored in the new triangle matrix in the slicing direction. Customize the slicing planes of equal thickness, ordered from bottom to top as an array slices \([\text{low}, \text{...}, \text{high}]\).

Step 2, each new triangular facet \( z_{\text{min}} \) stored in the node matrix, marking them. Set the boundary initial value of the binary search, so \( \text{low} = 1, \text{high} = \text{size(slices,2)} \), then \( \text{mid} = \text{low} + \text{floor}(\text{high} - \text{low})/2 \).

Step 3, using the binary search algorithm to match \( z_{\text{min}} \) of triangular facet with \( \text{slice(mid)} \) of slice plane. When \( z_{\text{min}} \) satisfies both \( \text{slice(mid)} > z_{\text{min}} \) and \( \text{slice(mid-1)} < z_{\text{min}} \), existence of
$z_{\text{min}} = \text{mid}$, otherwise do not match. If there exists $z_{\text{slice}}(\text{mid}) > z_{\text{min}}$, the triangular facets have intersecting slice planes in the array $z_{\text{slice}}[\text{low}, ..., \text{mid}-1]$. If there exists $z_{\text{slice}}(\text{mid}) < z_{\text{min}}$, the triangular facets have intersecting slice planes in the array $z_{\text{slice}}[\text{mid}+1, ..., \text{high}]$. Repeatedly search, each search reduces half of the data, the index is $z_{\text{low\_index}}$.

Step 4, to avoid matching errors, check the high with the binary search. The principle is the same as step 2. Extract the $z_{\text{max}}$ of each new triangular facet and update the node matrix. Set the boundary initial value, so that $\text{low} = 1$, $\text{high} = \text{size}(\text{slices}, 2)$.

Step 5, search for the $z_{\text{max}}$ and $z_{\text{slice}}(\text{mid})$ matching of the triangular facet, the same principle and step 3, the Index is $z_{\text{high\_index}}$.

Through step 3 to step 5, as long as the slice plane and the triangle face satisfy $z_{\text{min}} < z_{\text{slice}}(\text{mid}) < z_{\text{max}}$, the triangular facet belongs to the set of triangular facets intersecting the slice plane.

Step 6, when the array is $\text{high} > \text{low}$, going to return to step 2, until the index $z_{\text{high\_index}} = z_{\text{low\_index}}$. Then all slice planes and triangular facets are matched, that is, grouping matrix is build. Exit search.

According to the binary search algorithm, the triangular facets are intersected with the current slice plane, and the triangular facets are dynamically indexed into the grouping matrix. The grouping matrix is updated for each additional slice thickness. In building the grouping matrix, if there is a triangular facet that does not intersect with all the slice planes or if the triangular facet is in the slice plane, the triangular facet can be removed from the grouping matrix, thus reducing redundant data processing [8].

4. Generate contour lines

4.1. Intersection of triangular facets and slice plane
We have previously established a grouping matrix for triangular facets. In the set of triangular facet, the intersections of the triangular facets and the line segment created by the intersection points are determined according to the order of their intersection with the slice plane. In order to determine the intersection with the slice plane, we first need to judge which two sides of the triangular facet intersect with the slice plane. Assuming the triangular facet intersecting with the slice plane, the coordinates of the three vertexes are respectively $P_1(x_1, y_1, z_1), P_2(x_2, y_2, z_2)$ and $P_3(x_3, y_3, z_3)$. $Z_p$ is slice plane that intersecting with the triangular facets. The intersection of $t_1, t_2$ and $t_3$ can be obtained according to the linear equation. Then, the two intersection points between the plane and the intersecting triangular facets are determined, and the two intersection points are connected to form a contour line [9].

The slicing plane and the intersection of triangular facets determine the relationship between line segments shown in figure 3: If $z_{\text{min}}(P_2, P_3) < Z_p < z_{\text{max}}(P_2, P_3)$, then the slice plane intersects with the triangular facet on two sides A and B, which intersect the line segment is $L[t_1, t_3]$, as shown in case1 of figure 3. If $z_{\text{min}}(P_1, P_3) < Z_p < z_{\text{max}}(P_1, P_3)$, then the slice plane intersects with the triangular facets on two sides of B and C, and the intersection line is $L[t_2, t_3]$, as shown in case2 in figure 3; if $z_{\text{min}}(P_1, P_2) < Z_p < z_{\text{max}}(P_1, P_2)$, then the slice plane intersects with the triangular facets on two sides of A and C, and the intersection line is $L[t_1, t_3]$, as shown in case3 in figure 3.

![Figure 3. The intersection relation between slice plane and triangular facet.](image-url)
4.2. Output contour print path
After the intersecting line segments are obtained in turn, we need to link them in an orderly manner and convert them into closed print contour paths. Find the first triangular facet that intersects with the slice plane according to the set of triangular facet, solve the intersection node. Adjacency relationship between triangular facets, search for other intersected triangular facet nodes in turn. Correct contour lines need to satisfy the conditions of closure and non-self-compatibility. For non-standard contour lines, that is, some invalid connections when inter-node intersections are connected, the contour lines are interrupted or there are many repeated points on the contour lines. Contour lines need to be repaired or removed redundant information [10]. The specific steps are as follows:

Step 1, set an error precision between the nodes, this paper take $tol = 1e^{-8}$.

Step 2, the intersection of the triangular facets and the slice plane is stored in the node matrix, search for the node allocation, the redundant nodes are removed in the error precision range. Reorder the nodes according to the adjacency relationship between the meshes.

Step 3, connect the line segments between the nodes of the intersected triangular facets. Check whether the start and end nodes of the line segment are within the nodes error range of the node matrix. If there is true, connect the line segment, otherwise it is not connected.

Step 4, final check, for the start and end node numbers of intersecting line segments, connect two adjacent nodes.

Then use the depth-first search algorithm to create a continuous path between intersecting line segments of the graph [11]. According to the adjacency between the triangle facets mesh, starting from the first component of the start_nodes, access to its adjacent node node1, and then starting from node1, search node1 adjacent but not visited node2, and then starting from node2, make similar search visits in turn. Using the depth-first search algorithm, stop the search until it return to the start_nodes of the first component. In this way, all the slice plane contour lines are stored in the list in turn to complete the STL model slicing.

5. Application algorithm instance analysis
In the rapid prototyping, the slicing algorithm has higher stability. In order to verify the stability of the algorithm, Figure 4 shows three-dimensional model of the whale, the number of triangular facets converted from 3D to STL model is 322234. Figure 5 shows the STL model contour output using the algorithm of this paper, and the slice thickness chosen for clarity of the slicing effect is 0.8 mm, the number of slicing planes is 71.

In order to prove the efficiency of the slicing algorithm, this paper uses the software of Repetier-host integrated with slicing, part positioning and machine control functions for comparison. As can be seen from the time-consuming experimental results for the number of triangular facets and the slice in table 1 below, the algorithm involved in this paper is many times faster than the Slic3r and CurEngine slices built into the Repetier-host 3D print control software, processing tens of thousands of triangles per second. The increased slicing speed of the STL model allows the complex STL model to slice in less than a second. The slicing algorithm of this paper is obviously superior to Slic3r and CurEngine slices, that is, the practicality and efficiency of this algorithm is very valuable. Figure 6 is time-consuming line chart of the algorithm in table 1 with slic3r and CurEngine slices.
Figure 4. Three-dimensional model.

Figure 5. Contour after slicing STL model.

Table 1. The slice time comparison between this algorithm and the Slice3r and CurEngine.

| STL model     | Number of triangles | Slice thickness / mm | Slice layer number | Slice time / s |
|---------------|---------------------|----------------------|--------------------|----------------|
|               | Algorithm           | Slic3r               | CurEngine          |                |
| Piggy         | 3844                | 0.4                  | 49                 | 0.164          |
| Sphere        | 14396               | 0.4                  | 126                | 0.868          |
| Flower vase   | 23666               | 0.4                  | 421                | 1.214          |
| Person        | 36606               | 0.4                  | 247                | 1.825          |
| Bearing       | 1780                | 0.4                  | 121                | 0.490          |
| Cup           | 8682                | 0.4                  | 180                | 0.569          |
| Safety valve body | 3738            | 0.4                  | 193                | 0.855          |
| Valve cover   | 1342                | 0.4                  | 211                | 0.393          |
| Decelerator   | 2924                | 0.4                  | 74                 | 0.160          |
| Turbine       | 85052               | 0.4                  | 104                | 0.575          |

Figure 6. Table 1 algorithm of this article, Slic3r and CurEngine slice time-consuming line chart.

6. Conclusions
In this paper, a new slicing algorithm is proposed. The algorithm firstly uses the binary search and search slice plane to intersect with the triangular facet to build grouping matrix. Each triangle only traversal search, you can complete the slice plane intersection processing. The algorithm searches for time complexity $O(n \log k)$, where $n$ is the number of triangles and $k$ is the number of slice planes [12]. Compared with literature 12 (time complexity $O(n^2)$), this algorithm has obvious advantages in obtaining the contour line time-consuming, and improves the layering efficiency.

After the grouping matrix is established, according to the adjacency relationship between the triangular mesh sets, it is concluded that the intersection of the slice plane and the edge of the triangular mesh is an ordered sequence. Then the depth-first search algorithm is used to create continuous contours of nodes. The node search time complexity is $O(n^2)$, $n$ is the number of nodes, and the search intensity between nodes is increased.
Compared with the time-consuming experiments of slic3r and CurEngine slicing, this algorithm occupies a significant advantage in slicing time. Especially in related fields such as biomedical, jewelry, machinery manufacturing, etc., for the complex structure of the STL model, the model has a large number of triangular facets mesh, the intersection of slicing plane and the triangular facets. It improves the efficiency of slicing in the time, and has good practicability.

References
[1] Xiao R 2011 STL Files Reading Displaying and Manipulating  Journal of Ceramics 32(2) pp285-8
[2] Vatani M, Barazandeh F, Rahimi A R, et al. 2010 Improved slicing algorithm employing nearest distance method  Proceedings of the Institution of Mechanical Engineers Part B Journal of Engineering Manufacture 224(B5) 745-752
[3] Eragubi M 2016 Slicing 3D CAD model in STL format and laser path generation  International Journal of Advanced Manufacturing Technology 4(4) pp410-3
[4] Wang C and Li Z 2013 The Optimization and Application of the STL Model Layer Algorithm  Machinery Design & Manufacture 3 pp87-90
[5] Hou C C and Nan L 2014 Algorithm for rapid slicing in STL model based on grouping array  Manufacturing Automation 36(5) pp12-15
[6] Gregori R M M H , Volpato N , Minetto R and Silva M V G D 2014 Slicing Triangle Meshes: An Asymptotically Optimal Algorithm International Conference on Computational Science and ITS Applications: pp.252-5 IEEE Computer Society
[7] Jiang H and Wenqiang Y U 2016 Research on the method of adaptive slicing based on FDM  Manufacturing Technology & Machine Tool 10 pp38-43
[8] Luo N C, Jian X and Cui L 2009 Research and realization of a new improved binary search algorithm  Computer Era. 22(7) pp56-57
[9] Lin J Q and Sun C 2017 Research on Adaptive Slicing in Additive Manufacturing Technology  Machinery Design & Manufacture (6): pp70-3
[10] Zhou H and Wu J 2015 New optimization method of STL model slice contour on rapid prototyping  Manufacturing Automation 37(12) pp25-27
[11] Li C and Ueno M 2016 An extended depth-first search algorithm for optimal triangulation of bayesian networks  International Journal of Approximate Reasoning 80 pp294–312
[12] Su W, Liu H and Xinxiong A Z 2011 An algorithm for rapid slicing of STL model based on sorting by triangle adjacency in layers  Journal of Computer-Aided Design & Computer Graphics 23(4) pp600-6