THE SLICING ALGORITHM FOR FAST PROTOTYPE PRODUCTION HAS BEEN IMPROVED AND OPTIMISED

[1]Mr. S. Suresh,[2]Mr. P. Shivaraj,[3]Mr. Kuna Naresh Raj

[123]Assistant Professor

Department Of Mechanical Engineering

Kshatriya College of Engineering

ABSTRACT:

An enhanced slicing algorithm is given as a solution to the problem of inefficiency that arises during the slicing phase of rapid prototyping that is based on STL models. The approach constructs the integral topology of STL models in advance by utilising a hash table. This makes it possible to acquire con- tours directly. The method then decreases the search range in slicing by building the slicing relation matrix, which may significantly reduce the time cost of slicing. The temporal complexity of the method has been shown to be roughly linear, according to the demonstrations. The approach has been shown to be effective and efficient via application examples, and the findings demonstrate that it performs better than other methods that are currently available, particularly when the STL model is either complicated or huge.

Rapid prototyping, slicing algorithms, STL models, topology reconstruction, and 3D printing are some of the keywords associated with this field.

Introduction

With the continuous development of science and tech- nology, some challenges are put forward in the field of the manufacturing industry. The need to shorten the processing cycle and improve the flexibility of produc- tion is increasing day by day. Traditional manufactur- ing methods can no longer meet the requirements. By contrast, Rapid Prototyping (RP) is a faster and more flexible manufacturing means. Based on the principle of layered manufacturing, the RP technology decom- poses the three dimension (3D) model into a series of section outlines, then takes the section shape as the fill- ing boundary, and fills and accumulates layer upon layer until the complete component is formed. The pro- cess of discretizing 3D model into two-dimensional sec- tion is the basis of the whole manufacturing process, which is one of the key problems to be solved in rapid prototyping.

Currently, most rapid prototyping software systems treat Standard Triangle Language (STL) files as inter- mediate process files. The STL file consists of a series of discrete triangles that simulate 3D entities and describe them as the normal vector of the triangle and the tri-coordinate list of X, Y, and Z coordinates. For the determined layering direction, the layering process is to intersect the model with a series of tangent planes perpendicular to the layering direction, obtain the inter- section points of the tangent planes with different layer- ing heights and triangles, and connect these intersection points in order to form a closed contour. Obviously, when STL files are very large, the layered processing process will inevitably cause a lot of time consumption. At present, there are many improved hierarchical

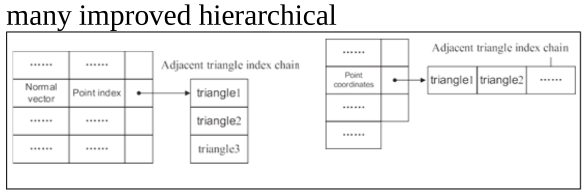


Figure 1. Data structure design: (a) point table data structure and (b) surface data structure.

processing algorithms, and the commonly used algo- rithms include hierarchical algorithms based on topol- ogy information.1,2 Hierarchical algorithm based on triangle position.3 Hierarchical algorithm based on tri- angle grouping.4,5 Hierarchical algorithm based on hash functions.6 Algorithm of coordinate layering based on STL model.7 Direct slicing algorithm based on compressed voxel model for rapid prototyping.8 Fast parallel algorithm based on pipeline mode.9 And the party check algorithms based on the fact that a straight line intersects any closed curve.10 However, for large or relatively complex models, the processing effi- ciency of most existing hierarchical algorithms is still unsatisfactory. In this paper, an improved hierarchical algorithm is proposed from the perspective of improv- ing the efficiency of hierarchical processing and com- bining the ideas of the algorithms in.1,4 In this algorithm, the topology reconstruction algorithm based on hash table is used to establish the overall topology structure of the model in advance, and then a hierarchi- cal relation matrix is established to solve the contour line. Through the comparison and analysis with the existing algorithms, it is proved that the algorithm in this paper significantly improves the layering efficiency.

Topological reconstruction based on STL model

Topology reconstruction algorithm based on hash search

Although the STL file is simple in structure and easy to handle, due to the organization of its data structure, each vertex of each triangle will be repeatedly recorded by other triangles sharing the vertex, resulting in the triangles recorded in the STL file being relatively inde- pendent of each other, and the lack of topological rela- tionships among the graphic elements (points, lines, and faces) constituting the entity, which greatly affects the efficiency of hierarchical processing.

Topological relation reconstruction of STL model mainly includes removal of redundant data and establishment of topological element adjacency rela- tion. The process of removing redundant data is the aggregation and merging process of vertices. Judging whether the new vertex is a redundant point by search- ing the vertex storage structure, thus ensuring that each vertex does not repeat recording. The process of estab- lishing the adjacency relation of geometric elements is to realize fast topological access between any point and edge, edge and face, face and face by establishing a rea- sonable data structure. The whole process of topology reconstruction is to continuously read triangles, remove redundant vertices by searching, establish a non- repeating point table, and record the inclusion and adjacency relationship between each topology element in the topology structure.

The geometric elements of STL model topology reconstruction mainly include points, edges and trian- gles. In order to reduce storage requirements, the topol- ogy structure designed in this paper only establishes point table and polygon table to record topology infor- mation between vertices and triangles. The data struc- ture of the design point table and polygon table is as follows in Figure 1:

For the more complex STL model, if the method of direct traversal is adopted in the process of finding ver- tices, the efficiency is undoubtedly extremely low. Many scholars have proposed solutions to the problem of excessive time cost in establishing topology structure. Basically, the vertex searching process is accelerated by introducing additional data structures. Zhang and Joshi11 introduces an improved and robust slicing algo- rithm with efficient contour construction to reduce the slicing time and be able to slice a complex STL model with millions of triangles (resulting from high-accuracy STL files). Jinyi and Baoming12 and Dai et al.13 intro- duces balanced binary tree to improve topological reconstruction efficiency. Zheng and Zheng14 and Tao et al.15 introduces the algorithm of topology reconstruc- tion combined with red-black tree structure. The time complexity of these two methods is O(nlgn), but in the case of a large amount of data, it takes a large time cost to maintain the balance of the tree structure. Wang16

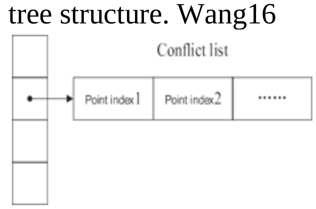


Figure 2. Data structure of Hash table.

and Yanyun et al.17 proposes to use Hash table as addi- tional data structure to quickly establish topology structure with time complexity up to O(n), but the space complexity is slightly higher than the previous two methods.

In this paper, a vertex hash table is constructed to realize fast topology reconstruction, and the Hash table loading factor is set to 0.5. Since the STL model is a polyhedron composed of triangles, it can be inferred from Euler formula that the actual number of vertices of the model is about one-half of the number of trian- gles, so the length of the hash table is taken as the largest prime number not greater than the number of triangles. According to the research conclusion of Skala et al.,18 after repeated tests, the Hash function of the following form is constructed:

$$k = floor((aV_x + bV_y + gV_z)K + 0{:}5$$ ð1 Þ
$$H(k) = |k\%p|$$

where floor means rounding down, % means remain-
der, Vx, Vy, and Vz are X, Y, and Z coordinates of vertices, a, b, g, and K are coefficients, and p is Hash table length.

In the actual application process, it is difficult to ensure that each hash address only corresponds to a unique key value, which will result in hash conflicts. In the chain address method, all elements with hash address ''i'' form a linked list called synonym chain, and the head pointer of the linked list is stored in the ith unit of the hash table, so the search, insertion and deletion are mainly carried out in synonym chain. In this method, firstly, a hash function is used to calculate the storage location of key sets. If all positions in the address space of the hash table are from 0 to m–1, all the keys in the key set are divided into m subsets, and the keys with the same address belong to the same sub- set. We call the keys in the same subset synonymous with each other. Each subset is called a bucket. Usually, the entries in each bucket are linked by a linked list,a vector. In this paper, chain address method is used to resolve hash conflicts, and key values with the same hash address are added to the linked list of correspond- ing addresses. The structure is shown in Figure 2.

The complete topology reconstruction process is as follows:

(1) Establishing a vertex Hash table, a point table and a surface table;

(2) Reading in a triangle;

(3) Calculating Hash addresses of three vertexes;

(4) Searching the corresponding Hash address in the vertex hash table, if so, indicating that the point already exists, and turning to step 5; If not found, it means a new point, go to step 6;

(5) The point already exists, add the index of the current triangle to the adjacent triangle index chain of the vertex in the point table, and go to step 7;

(6) The point is a new point, adding the index of the current triangle to the index chain of the triangle adjacent to the point, adding the point to the point table, and adding the index of the point to the corresponding address in the Hash table;

(7) Adding the indexes of the three vertices in the point table to the current triangle, and adding the triangle to the surface table;

(8) Go to step 2 until all triangles have been read.

According to the topological relation established
above, the topological structure between points and planes can be efficiently established, which provides necessary preconditions for subsequent data processing.

Analysis of algorithm efficiency

In this algorithm, Hash table is used to speed up the process of topology reconstruction. Hash table can directly access a data structure of the specific corre- sponding value through the given key value, and map the key to the position in a table to directly access the record, thus speeding up the access speed, thus acceler- ating the speed of topology reconstruction, but the effi- ciency of Hash table depends heavily on the performance of Hash function. The design of hash function directly determines the performance of hash table by determining the collision probability of hash table. Therefore, if the hash function is not too complicated or the loading factor is too high, the probability of hash collision will increase, and the query efficiency will decrease. Therefore, a reasonable Hash function is the main factor that determines the efficiency of the algorpithffiffiffim. The parameters are chosen same bucket number are linked in the same synonym sub-table, and the header nodes of each linked list form different sizes, and the distribution trend of Hash table's maximum bucket length is shown in Figure 3:

as a = p, b = e, g = 2, $K = 10^6$, to test STL files of

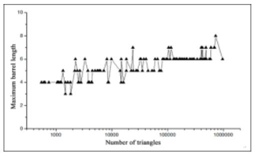


Figure 3. Maximum bucket lengths of hash table.

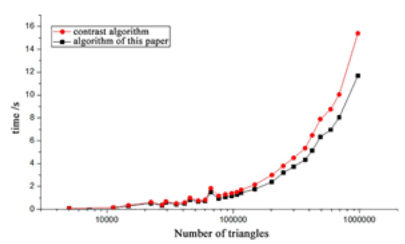


Figure 4. Time consuming comparison of different algorithms.

The test results show that the Hash function designed in this paper can well disperse the model ver- tices, and the maximum barrel length can be stabilized within a certain range for STL files of different sizes, which proves that the Hash function in this paper has good performance.

Taking topology reconstruction algorithm based on red-black tree as comparison algorithm, STL files of different sizes are tested under the same software and hardware environment. The results are as follows in Figure 4:

The results show that compared with the existing better algorithms, the algorithm adopted in this paper effectively reduces the time for topology reconstruction. When the STL file is large, the algorithm in this paper shows more obvious advantages.

Improved stratification algorithm of STL model

Construction of hierarchical relation matrix

In order to reduce the judgment of intersection relation, it is first necessary to establish the hierarchical relation matrix of tangent plane and triangle. The hierarchical relation matrix uses the

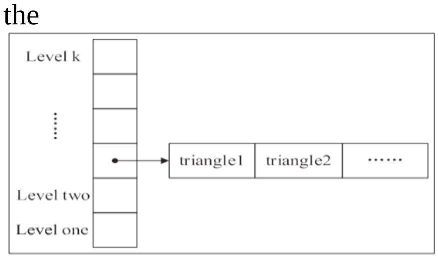


Figure 5. Slicing relation matrix.

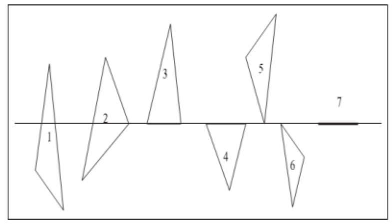


Figure 6. The position relationship of triangles and the tangent plane.

serial number of each hierarchi- cal layer as an index, and adds all triangles that have intersection points with the same hierarchical layer to the same layer of the matrix, thus establishing a set of intersecting triangles for each hierarchical layer. Its data structure is shown in Figure 5.

In the actual situation, the positional relationship between the triangle and the sub-level is relatively compli- cated, and there is a case where the triangle intersects the sub-level but the intersection point does not need to be calculated. Therefore, in the grouping process, it is neces- sary to specifically analyze the relative positional relation- ship between triangles and tangent planes, and to exclude the situation that intersection points do not need to be calculated.

In the case where the triangle intersects the tangent plane, the positional relationship can be divided into the following seven types as shown in Figure 6:

For cases 1 and 2, triangles pass through the tangent plane. These two cases will not produce repeated inter- section points, and both need to be calculated. For cases 3 and 4, the triangle intersects the tangent plane on one side. According to the common-edge rule of STL model, there must be an adjacent triangle inter- secting the plane on the same side in both cases. If the intersection point is calculated for both cases, the inter- section point will be repeatedly calculated, which will affect the generation of contour lines. Therefore, only the triangle conforming to case 3 will be retained here, and case 4 will be eliminated. For cases 5 and 6, the tri- angle intersects the tangent plane at one point. From

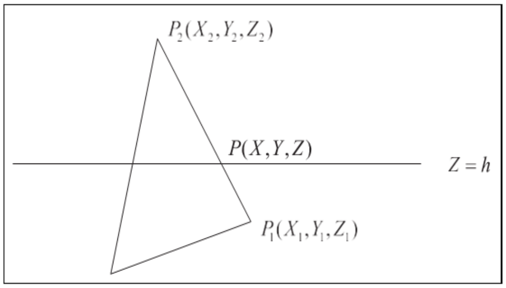


Figure 7. Calculation of the intersection point of the edge and the tangent plane.

the closeness of STL model, it can be seen that among the adjacent triangles located at the vertices on the tan- gent plane, there must be an adjacent triangle intersect- ing the tangent plane at two points. For the last case, the triangle lies on the tangent plane. According to the closeness of STL model, a group of triangles adjacent to the triangle intersect with the tangent plane on one side, so this situation can also be ignored. To sum up, in the process of stratification, only the first three situa- tions need to be considered.

According to the above analysis, the triangular hier-
archical interval division rules are established as follows:

Assuming that the layering direction is the positive direction of the Z axis, the height of the layer is h, the initial layering height is Z0, and the three vertices of the triangle are ZMax, ZMid, and ZMin respectively accord- ing to the size of the Z coordinate, then the layering sequence number interval (m, n) intersecting with the triangle can be calculated by the following formula:

$$\begin{cases} m = ceil((Z_{Min} - Z_0)/h) \\ n = ceil((Z_{Max} - Z_0)/h) \end{cases} \quad (2)$$

where ceil indicates an upward integer value. Considering the following special cases:

(1) if the triangle satisfies $Z_{Min} = Z_{Max}$, skip the triangle;
(2) if the triangle satisfies $Z_{Max} <= Z_0$, then the triangle is not considered;
(3) if the triangle satisfies $Z_{Min} < Z_0$, then the corresponding interval of the triangle is $(0, n)$;
(4) if the triangle satisfies $Z_{Min} = Z_0$ and $Z_{Min}! = Z_{Mid}$, then the corresponding interval of this triangle is $(m + 1, n)$.

According to the above rules, the establishment of hierarchical relation matrix can be completed by sequentially accessing each triangle, calculating its hierarchical interval and adding an index to each group corresponding to the interval.

Searching for intersections on opposite sides

After obtaining the hierarchical relation matrix, the contour line can be calculated by topological relation according to the triangles recorded in the matrix. For each sub-level, the corresponding triangle grouping is taken out, the adjacent triangles are continuously tracked through topological relation and the intersec- tion points are calculated one by one. When all trian- gles in the group are accessed, the slice outline of this layer has been extracted.

Calculating the intersection of a triangle and a sub- division is essentially calculating the intersection of a triangle's edge and a subdivision. According to the par- tition rule proposed in the previous section, it can be seen that the triangles retained in the hierarchical matrix all intersect the tangent plane at two points. Obviously, if the intersection points are calculated in triangular units, then each intersection point will be calculated twice. Therefore, the intersection point of contour lines is calculated based on edges to avoid duplication of intersection data.

The location of the intersection between the edge and the stratification plane is relatively simple, either passing through the plane or crossing at an endpoint. Here, only the intersection point needs to be calculated for the first case. The schematic diagram of the inter- section between the edge and the stratification plane is as follows in Figure 7:

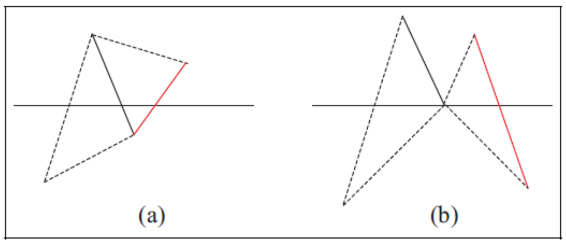
According to the spatial geometric relations, there are:

$$\frac{X - X_1}{X_2 - X_1} = \frac{Y - Y_1}{Y_2 - X_1} = \frac{Z - Z_1}{Z_2 - Z_1} \quad (3)$$

The obtained intersection coordinates:

$$\begin{cases} X = \dfrac{Z - Z_1}{Z_2 - Z_1}(X_2 - X_1) + X_1 \\ Y = \dfrac{Z - Z_1}{Z_2 - Z_1}(Y_2 - Y_1) + Y_1 \\ Z = h \end{cases} \quad (4)$$

For a triangle, the value of Z_{Mid} determines the edge that intersects the facet. If Z_{Mid} is just above or below the sub-level, the intersecting edges are two edges formed by Z_{Min}, Z_{Max} and Z_{Mid}, Z_{Max}. On the other hand, if Z_{Mid} is located above the stratification plane, the intersecting edges are Z_{Min}, Z_{Max} and Z_{Mid}, Z_{Max}. In order to prevent triangles from repeated access, a flag bit is set for each triangle. When the triangle is accessed, it is set. The flag triangle has been processed.



When the contour calculation of one layer is finished, reset it for the contour calculation of the next layer. Defining the search rules:

(1) as shown in Figure 8a, if the current edge passes through the tangent plane, the triangle sharing this edge is directly taken out from the current triangle as the triangle to be accessed, and the unprocessed edge with intersection point is taken as the next edge to be processed;

(2) as shown in Figure 8b, if the current edge inter- sects the tangent plane at an endpoint (a vertex of the triangle), in this case, the adjacent trian- gle of this vertex is found. Since the current tri- angle has been marked at this time, only an adjacent triangle that meets the requirements and the flag bit is not set can be found, then this triangle is the next triangle to be accessed. Obviously, this triangle intersects the current triangle at this vertex, and the edge formed by the other two vertices is taken as the next edge to be processed.

Since the STL model is a closed entity, the cross- sectional profile of each layer must also be closed. Search according to the above rules. If the next unmarked triangle cannot be searched, then the contour has been closed, that is, it returns to the origi- nal triangle, thus a contour has been formed and the search process is finished. However, if there are multi- ple closed contours in one layer, there will be unpro- cessed triangles in the triangle group recorded in the hierarchical matrix. Therefore, after generating a con- tour line, it should also be judged whether there are tri- angles that intersect the layer but have not been processed: if there are such triangles, the new contour line should be calculated continuously. If all the inter- secting triangles have been processed, the contour lines of this layer have been extracted.

To sum up, the main process for extracting contour lines is as follows:

(1) Creating a new contour line;

(2) Taking any triangle as a starting triangle in the triangle grouping of the current layer, and tak- ing out an edge intersecting with the sub- layer as an edge to be processed;

(3) Judging the positional relationship between the edge to be processed and the stratification plane. If it is handed over to the endpoint, record the point directly. Otherwise, the inter- section point is calculated according to formula(4)and the contour line is added;

(4) Setting that current triangle mark position bit;

(5) Searching for the next triangle to be accessed according to the search rule, if the next triangle can be searched, finding the next side to be pro- cessed, and turning to step 3. If not, go to step 6.

(6) Judging whether unmarked triangles exist in the triangle grouping. If it exists, take this triangle as the starting point of the new contour line and go to step 1. If it does not exist, go to step 7.

(7) Reset the flag bits of all triangles in the group- ing, and end the contour extraction of the current layer.

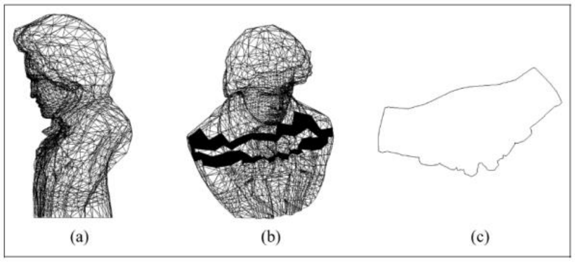
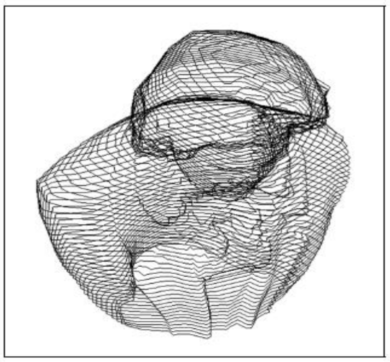


Figure 9. The slicing process of Beethoven model: (a) Beethoven model, (b) set of intersecting triangles, and (c) contour line.



Analysis of algorithm efficiency and case verification

Analysis of algorithm efficiency

The hierarchical algorithm proposed in this paper first constructs a hierarchical relation matrix, then calcu- lates contours hierarchically using global topological relations, and finally obtains a complete set of con- tours. Compared with the general hierarchical algo- rithm based on topological relations, the algorithm in this paper groups triangles in advance and excludes the situations that do not need to be considered. During the hierarchical process, the intersection triangle set can be directly obtained, which reduces the computa- tion of intersection judgment and avoids the occurrence of invalid situations. Compared with the layered algo- rithm based on triangle grouping, the algorithm in this paper establishes the global topology structure in advance by optimization method, which avoids the time consuming of establishing the local topology structure. For different layered thicknesses, the topol- ogy structure only needs to be established once without repeated calculation.

Considering the hierarchical process proposed in this paper, it is mainly divided into three stages: topology reconstruction, triangle grouping and contour calcula- tion. Let the number of triangles be N and the number of layers be k, with each triangle intersecting with l sub-layers on average, and each sub-layer intersecting with f triangles on average. In the process of topology reconstruction, each triangle is processed once, and the time complexity of hash lookup can be approximately considered as O(1), so the overall time complexity is O(N). In the grouping process, all N triangles need to be traversed, each triangle performs l grouping operations, and the overall time complexity is O(l———— N). In the worst case, each triangle intersects all k sub-planes, where l=k. In general, k is much less than N, so the complexity is approximately O(N). In the process of

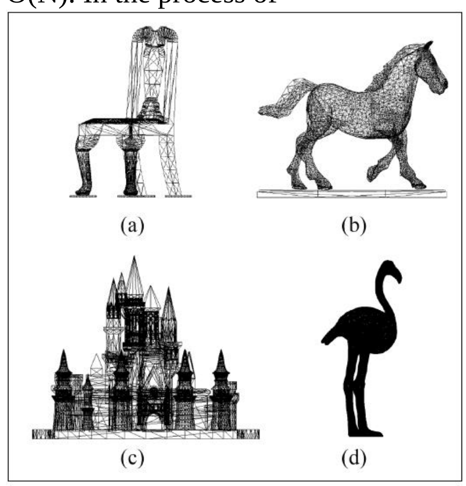


Figure 11. Test models (a) the number of model triangles with 6226, (b) the number of model triangles with 11244, (c) the number of model triangles with 45486, and (d) the number of model triangles with 104270.

calculating the contour line, a total of k packets need to be traversed, and each packet needs to access all f elements in the packet, so the time complexity is O(k f). Considering the worst case, each sub-level passes through all N triangles, and the complexity becomes O(k N), which can be approximately considered as O(N). According to the above processes, the overall time complexity of the hierarchical algorithm is O(N), and the overall efficiency of the algorithm is very high.

Application case verification

Taking Beethoven model as an example, the triangle set and contour line corresponding to one layer are shown in Figure 9:

The overall layering effect of the model is shown in Figure 10:

According to the above results, it can be seen that the algorithm can correctly divide triangles and gener- ate complete contour lines, which verifies the correct- ness of the algorithm.

Four models shown in Figure 11 are selected as test examples, and the number of triangles in each model is 6226, 11,244, 45,486, and 104,270 respectively. Under different layering thicknesses, the layering algorithm based on topological relation (algorithm 1), the layering algorithm based on triangle grouping (algorithm 2) and the algorithm proposed in this paper are used to stratify each model respectively, among which algorithm 1 uses the method introduced in this paper to establish

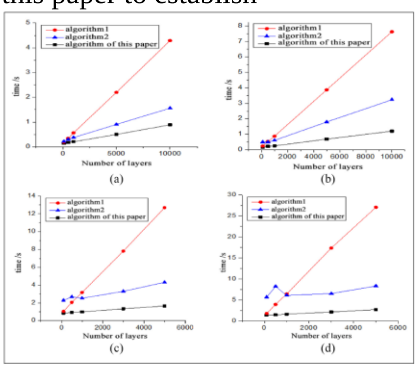


Figure 12. Time consuming comparison of different algorithms with models in Figure 11(a–d).

topological structure, From the experimental results, the algorithm used in this paper can improve the effi- ciency by 2–7 times compared with the algorithms in different layers. The required time comparison is as follows:

Compared with other algorithms, this algorithm has a smaller time cost (Figure 12). Take Figure 12c as an example. When the number of layers is 3000, algorithm 1 takes 8 s, algorithm 2 takes 3 s, and the algorithm used in this paper takes 1 s, which is several times higher than other algorithms. The results show that, in the case of large STL files, the time consumption of the algorithm in this paper is greatly reduced compared with the other two methods, which proves the efficiency of the algorithm in this paper.

Conclusion

In order to solve the problem of low efficiency of exist- ing layering algorithms, this paper proposes an improved STL model layering algorithm. The algo- rithm uses the topology reconstruction algorithm based on hash search to establish the overall topology struc- ture of the model in advance, and then reduces the search scope of hierarchical calculation by establishing hierarchical relation matrix, thus effectively reducing the time consumption of STL model layering.

Through the example verification, it is proved that the algorithm can correctly generate the cross-sectional profile and effectively improve the layering efficiency. Compared with other algorithms, this algorithm has less time cost, especially for the more complex STL model, this algorithm has more obvious advantages.

Acknowledgement

We thank Prof. Chengwei Li (School of Electrical Engineering and Automation, Harbin Institute of Technology) for the help in conducting this experiment.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial sup- port for the research, authorship, and/or publication of this arti- cle: The work was supported by General Program of Science and Technology Development Project of Beijing Municipal Education Commission (Grant No. KM202310005034), Beijing Science and Technology Planning Project (Grant No. Z221100006422007), and Beijing Postdoctoral Research Foundation (Grant No. 2022-zz-056).