


Article

Delaunay Mesh Construction and Simplification with Feature Preserving Based on Minimal Volume Destruction

Yu Huo ^{1,2,3} , Tongcai Wang ^{1,2,3,*}, Haochen Li ^{1,3}, Yu Zhang ^{1,2,3}, Xin Li ^{1,2,3}, Bingshan Liu ^{1,2,3} and Gong Wang ^{1,2,3,*}

¹ Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences, No. 9 Dengzhuang South Road, Beijing 100094, China; huoyu16@csu.ac.cn (Y.H.); lihaochen@csu.ac.cn (H.L.); zhangyupaul0228@gmail.com (Y.Z.); lixin@csu.ac.cn (X.L.); liubingshan@csu.ac.cn (B.L.)

² University of Chinese Academy of Sciences, Beijing 100049, China

³ CAS Key Laboratory of Space Manufacturing Technology, Beijing 100094, China

* Correspondence: wangtongcai@csu.ac.cn (T.W.); wanggong@csu.ac.cn (G.W.)

Abstract: Triangular meshes play critical roles in many applications, such as numerical simulation and additive manufacturing. However, the triangular meshes transformed from computer-aided design models using common algorithms may have many undesirable narrow triangles, which tends to affect the downstream applications. In this paper, we proposed two algorithms for Delaunay mesh construction and simplification to improve the quality of the triangular meshes. Two improved mesh operations of inserting vertices and collapsing vertices based on the principle of minimum volume destruction were designed. The improved vertex inserting operation is able to modify the local mesh so that it will conform to the local Delaunay property. The improved vertex collapsing operation can realize the simplification of the original mesh while maintaining the local Delaunay property. The results of visualized rendering and thermal diffusion simulations verified the improvement of the proposed algorithms in the aspects of the quantity and quality of the meshes.



Citation: Huo, Y.; Wang, T.; Li, H.; Zhang, Y.; Li, X.; Liu, B.; Wang, G. Delaunay Mesh Construction and Simplification with Feature Preserving Based on Minimal Volume Destruction. *Appl. Sci.* **2022**, *12*, 1831. <https://doi.org/10.3390/app12041831>

Academic Editor: Giancarlo Mauri

Received: 29 December 2021

Accepted: 8 February 2022

Published: 10 February 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: Delaunay mesh; construction; simplification; geodesic distance

1. Introduction

Computer-aided design (CAD) models have played crucial roles in technical fields, such as numerical simulation, additive manufacturing and visualization. Accordingly, the triangular mesh, generated from CAD models, has become the dominant type of functional element due to its advantages of efficiency, simplicity and flexibility. However, triangular meshes, which are always transformed directly from a CAD model, usually have the defects of excessively small or large angles, and redundant or irregular vertices. These defects tend to reduce the accuracy of the numerical simulation based on triangular meshes. Thus, it is necessary to establish effective remeshing algorithms to optimize the generated triangular meshes, which is important for the downstream applications.

Alliez et al. [1] summarized many theories and applications of remeshing algorithms, which can be divided into the quantity-dominant simplification algorithms and quality-dominant optimization algorithms. In the aspects of simplification algorithms, Hoppe et al. [2] presented a series of basic mesh operations for mesh simplification. Garland and Heckbert [3] also proposed a kind of quadratic error metrics (QEM) simplification method, which is also regarded as the general model simplification algorithm. As for the optimization algorithms, Botsch and Konnelt [4] first introduced the Laplace–Beltrami (LB) operator to improve the quality of the irregular triangular meshes. Based on the LB operator, they were able to construct area-equalizing triangulations with vertex smoothing operation and optimize the original mesh, which was regarded as the basic optimization algorithms. Furthermore, Wang et al. [5] transformed the original judgment condition for edges into the angles, enabling the simplification after the mesh optimization. Wang et al. [6] have realized the constraint of the shape of the triangle by varying the size of bubble based

on the dynamics simulation method. Khan et al. [7] also tried to plan a more reasonable distribution of the number of vertex-to-vertex connections and optimize the meshes by constraining the range of angles. Most of these algorithms were based on the method of area-equalizing triangulation, which were used to achieve optimization and simplification of meshes in forms of regular triangles. Although the regular triangles can improve the computational accuracy of simulation results, the overemphasis on the evenly constructed triangular elements may limit the reduction of data storage and cause destruction to the geometric feature information of the original model. It is considered that the accuracy of numerical simulations can be improved when the cotangent formula $\cot \alpha + \cot \beta$ of the discrete LB operators has non-negative weight. According to this theory, a geodesic distance based on the discrete LB operator was used to simulate the thermal diffusion by Crane et al. [8]. Moreover, Bobenko and Springborn [9] proved that the Delaunay triangulation could always satisfy the non-negative weight for discrete LB operators, which was defined as the local property of the Delaunay triangulation. Correspondingly, the global property of the Delaunay triangulation was defined as the maximization of all the minimum angles in the triangulation, which can be used to indirectly constrain the shape of the triangle and avoid narrow triangles [10]. It is obvious that the Delaunay triangulation has the advantages in protecting the geometric features and simplifying the original meshes. However, it is difficult to achieve the Delaunay triangulation on the Riemannian manifold, considering that the geodesic paths are difficult to be directly represented with existing data structures. To solve this problem, Dyer et al. [11] defined a special concept of Delaunay mesh that, as long as the sampling criterion acquired a relatively lower bound among the distances of vertices in the Voronoi diagram, the Delaunay mesh can be obtained. Both Dyer et al. [12] and Liu et al. [13] proposed their adaptive sampling criterions to convert an arbitrary mesh into the Delaunay mesh by splitting non-Delaunay edges. Their methods tended to increase the number of vertices while the original mesh was converted to the Delaunay mesh. Therefore, the computational complexity including space complexity of data storage and time complexity of model processing will be inevitably increased, which may limit the application and development of Delaunay triangulation, especially for the case of original meshes with narrow triangles. Although Yi et al. [14] proved a hybrid mechanism operation of edge splitting and collapsing based on differential equations, which were able to reduce the complexity of the computational processing, the basic principle of algorithms has not been optimized.

In this paper, we propose two algorithms for the construction and simplification of Delaunay meshes. The Delaunay construction algorithm can transform arbitrary triangular meshes into the Delaunay meshes with the same topological type. Because the positions of the inserted vertices are calculated according to the geometric information of the mesh, and the identification criteria of the narrow triangles are designed to terminate the algorithm loop, both the space complexity and the time complexity of the Delaunay mesh construction algorithm can be optimized. In addition, a simplification algorithm is designed to simplify the inserted vertices during the construction process, and then it will decrease the data storage of the constructed meshes. The hybrid algorithms can preserve the geometric features of the original CAD model with the principle of minimal volume destruction. The experiments show that the proposed algorithms were effective for optimizing the non-Delaunay meshes, and especially for the improvement of the quality and quantity of the Delaunay meshes.

2. Related Work

2.1. Definition of Delaunay Triangulation

The Delaunay triangulation has similar effects in the numerical simulation with the regular triangulation. It is defined as its duality with the Voronoi diagram, which is shown as the dashed lines in Figure 1. The region of the Voronoi diagram R_i is divided according to the nearest neighbor principle, where point is associated with its nearest neighbor region. R_i can be expressed as the following:

$$R_i = \{p | d(p, v_i) \leq d(p, v_j), j = 0, 1, 2, \dots, n, i \neq j\} \quad (1)$$

where p is any point in the defined space and $d(p, v_i)$ is the distance between p and v_i . In particular, as the dual graph with the Voronoi diagram, the Delaunay triangulation mesh has the local property, which is also named as the empty geodesic circumcircle property. This means that, for a triangle t consisting of vertices v_0, v_1, v_2 in the Delaunay triangulation mesh M , no other vertices that are not the vertex of the triangle in the circumcircle of t exist. The local property implies that the commonly used cotangent Laplace–Beltrami operator has non-negative weights [13], which enables the Delaunay triangulation to have the similar performance as the area-equalizing triangulation in the numerical simulation.

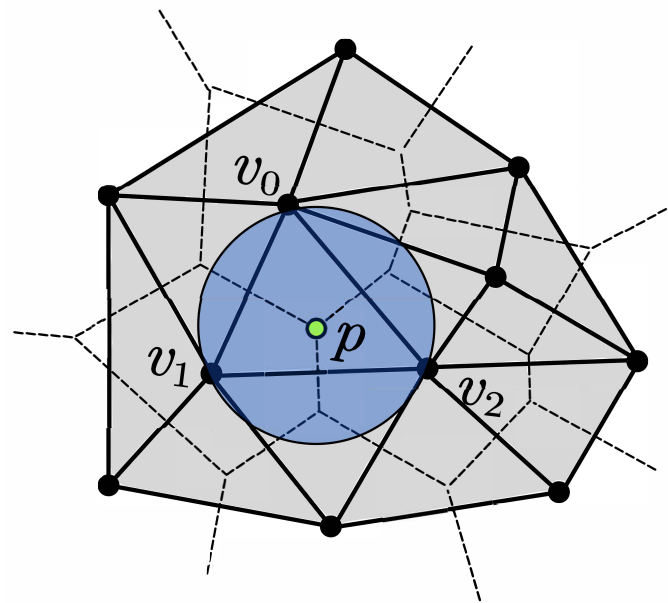


Figure 1. The illustration of the local Delaunay property.

2.2. Intrinsic Delaunay Triangulation

Rivin et al. [15] first defined the intrinsic Delaunay Triangulation on the manifold surface. Then, Leibon et al. [16] proved that a unique Delaunay triangulation existed on each closed Riemannian manifold, but it needs a large number of vertices to construct Delaunay triangulation this way. Since the geodesic paths, also known as the triangular edges of iDT, do not uniquely exist on the Riemannian manifold, it is necessary to confine the domain to a sufficiently small area by vertices. Chen et al. [17] presented a two-dimensional Delaunay mesh smoother based on the optimal Delaunay triangulation (ODT), and they also tried to extend the ODT to three-dimensional conditions. However, on the Riemannian manifold, it is not easy to directly transform arbitrary triangle mesh into the Delaunay mesh. Because the geodesic paths are difficult to directly represent by existing data structures, one indirect method was proposed by using a Geometric Voronoi Diagram (GVD) [18,19], namely the dual of the Delaunay triangle, as an intermediate element. Liu et al. [20] proposed a Centroidal Voronoi Tessellation (CVT) approach that was proved to be able to construct high quality meshes. The classical Lloyd's iterations for CVT computing are linearly convergent and inefficient. Then, some algorithms are built to improve the efficiency of this kind of indirect construction algorithm by increasing the speed of solving. Liu et al. [21] proposed a quasi-Newton method to accelerate the speed of convergence. Wang et al. [22] summarized and compared the aforementioned methods and the limited-memory BFGS (L-BFGS) method in detail. They also contrived a way to efficiently compute the Riemannian center on the basis of the discrete exponential map. Subsequently, Liu et al. [23] presented a novel method relying on manifold differential evolution (MDE), and obtained the globally optimal geodesic on triangle meshes.

3. Method

In this section, we first give the necessary symbol definitions and several standard local mesh operations. Then, two important improved mesh operations and the algorithm based on them are designed.

3.1. Symbol Definition and Standard Mesh Operations

Given an input mesh $M = (V, E, F)$ is a closed 2-manifold triangle mesh, where V, E, T are the sets of vertex, edge, and triangular element, respectively, for an edge e_{ij} , vertices (v_i, v_j) are the endpoints of it. The definition of the 1-ring neighborhood $\Omega(v_i)$ of $v_i \in V$ is shown in Figure 2a:

- For vertex v , if the $v \in \Omega(v_i)$, the set $\{v\}$ is all of the vertices adjacent to v_i , as shown in red vertices in Figure 2a.
- For edge e_{ij} , if $e_{ij} \in \Omega(v_i)$, the set $\{e_{ij} | 1 \leq i, j \leq n\}$ is all of the edges whose endpoints are the pair of vertices (v_i, v_j) .
- For triangle t_{ijk} , if $t_{ijk} \in \Omega(v_i)$, the set $\{t_{ijk} | 1 \leq i, j, k \leq n\}$ is all of the triangles incident to vertex v_i .
- In particular, for edge e_{jk} , if $e_{jk} \in t_{ijk}$ and $t_{ijk} \in \Omega(v_i)$, the set $D(v_i)$ could denote all e_{jk} which satisfy the condition, as indicated by the blue lines in Figure 2a.

Figure 2b shows a series of standard local mesh operations. These operations were originally designed for the mesh simplification algorithm [2]. The effect on the local Delaunay properties of the mesh is not considered when the operations are executed. It is necessary to improve them to make these operations more suitable for Delaunay mesh construction and simplification algorithms.

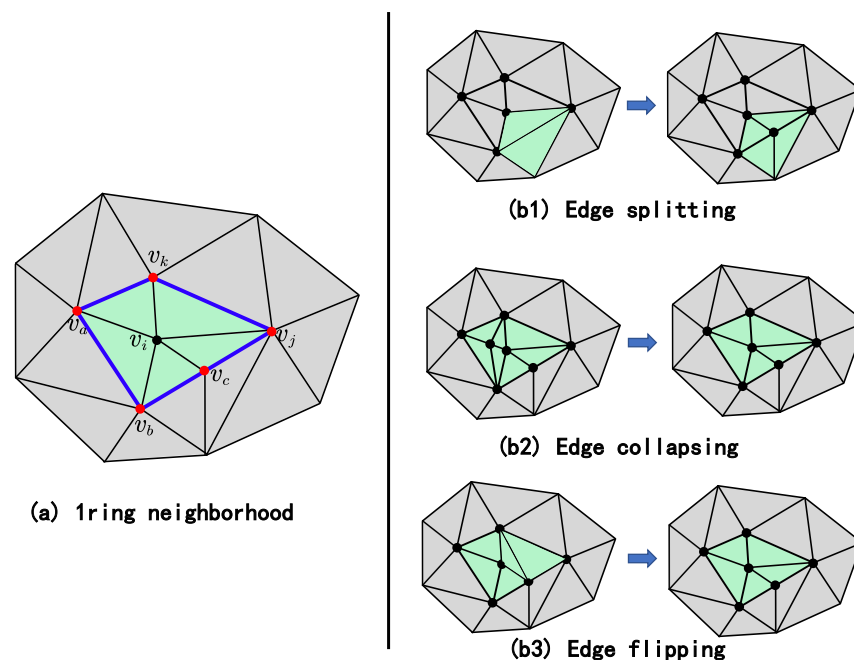


Figure 2. (a) Schematic of a 1-ring neighborhood $\Omega(v_i)$; (b) standard operations for the remeshing algorithm.

3.2. The Delaunay Mesh Construction Algorithm

The vertex inserting is considered as an improved version of the edge splitting. For a local non-Delaunay mesh M , it is possible to satisfy the local Delaunay property by choosing the appropriate position for inserting a new vertex. First, an assumption of a general situation and related inferences is given.

As shown in Figure 3, the local non-Delaunay mesh M consists of four vertices v_i, v_g, v_j and v_k . For a non-locally Delaunay edge e_{ij} , the sum of the two opposite angles $\angle v_i v_k v_j$

and $\angle v_i v_g v_j$ is greater than π . In this case, at least one of the two opposite angles of e_{ij} exists as an obtuse angle, and e_{ij} is the longest edge of this obtuse triangle t_{ijk} . Such that e_{ij} is not the shortest edge in the set $E_i = \{e_{ij} | e_{ij} \in \Omega(v_i)\}$ because there must exist both e_{ij} and e_{ik} belonging to $\Omega(v_i)$. Thus, we assume that edge $e_{ia} \in E_i$ is the shortest edge of E_i . As shown in Figure 3, the same length on e_{ij} is intercepted at point v'_a , and O is the midpoint of this length. Given any inserted vertex s on the line between O and v'_a , the intersection of the gray area with e_{ij} is the targeted range of s . Considering the relation of $l_{Av_i} = l_{sv_i} = l_{Bv_i}$, the suitable location can be determined by the following conditions:

$$\angle v_i v_k s + \angle v_i v_g s \leq \angle v_i A s + \angle v_i B s = \frac{2\pi - \angle v_k v_i v_g}{2} < \pi, \tag{2}$$

Similarly, we can also obtain a vertex s' for the case of v_j . If the regions of s and s' are overlapped, the inserted vertex is chosen as the midpoint in the regions. However, if these regions are not overlapped, the above process can be repeated and the convergence of it are acquired due to the monotonically of $sum_1 = \angle v_i v_k s + \angle v_i v_g s$ and $sum_2 = \angle s v_k v_j + \angle s v_g v_j$. As the region where s is taken gradually moves away from v_i to v_j , sum_1 monotonically increases and sum_2 monotonically decreases. Both of them are less than π , which means e_{is} and e_{sj} can fulfill the local Delaunay property. The improved vertex inserting operation can choose the suitable location for the newly inserted vertex s , and then the local Delaunay property can be achieved. After the improved vertex inserting operation, the Delaunay property of the region consisting of four vertices v_i, v_g, v_j and v_k is satisfied. These new angles $\angle v_i s v_k, \angle v_i s v_g$ and $\angle v_j s v_g$ destroy the local Delaunay property of the surrounding region, so the repeated vertex inserting is acceptable. However, in the case of narrow triangle appearance, it may repeat the vertex inserting operation endlessly. Therefore, a control parameter δ should be designed to detect narrow triangles and avoid this kind of useless repetitive operation.

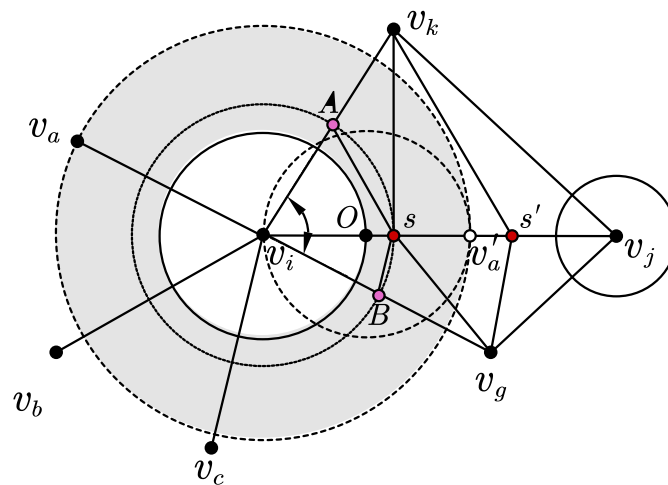


Figure 3. Choosing the position of newly insert vertex. For the original vertex v_i , the newly inserted vertex s on the line segment between O and v'_a . Similarly, for the v_j , an inserted vertex s' is been obtained.

For the original mesh M , l_{min} is the length of the shortest edge. θ_{min} is the smallest angle. h is the distance between vertex v_k and bottom edge e_{ij} of triangle t_{ijk} . d is the distance between vertex v_k and inserted vertex v_s . For a Delaunay mesh DM complied with the local Delaunay property, all the edges of DM satisfy the following conditions:

- Conditions 1: As shown in Figure 4a, for any triangle t_{ijk} in M , this state is always satisfied:

$$d \geq h = l \cdot \sin(\theta) \geq l_{min} \cdot \sin(\theta_{min}), \tag{3}$$

- Conditions 2: For original edge e in M , the condition is also found that:

$$\min\{l_e | e \in \Omega(v_i)\} \geq l_{min} \cdot \sin(\theta_{min}), \tag{4}$$

- Conditions 3: As shown in Figure 4b, the distance of the newly inserted vertex from any vertex also satisfies that:

$$l_{is} \geq l_{ig} \cdot \sin(\theta_{min}) \geq l_{min} \cdot \sin(\theta_{min}), \tag{5}$$

Therefore, if a new edge e_{sg} is shown in Figure 4b, whose length is less than $l_{min} \cdot \sin(\theta_0)$, the triangle t_{ijg} could be considered as a narrow triangle. In the practical application, the l_{min} and $\sin(\theta_{min})$ do not exist in the same triangle. Thus, a control factor δ could be set as $\delta = k \cdot l_{min} \cdot \sin(\theta_0), k \in \mathbb{Z}^+$ to replace $l_{min} \cdot \sin(\theta_0)$. In other words, if the length of a new edge e_{sg} is less than δ , the triangle will be considered as a narrow triangle which is collapsible in the algorithm and shown in Figure 4c. This condition is considered as the case of volume destruction to the original mesh because it makes the final narrow triangle compressed into a single edge.

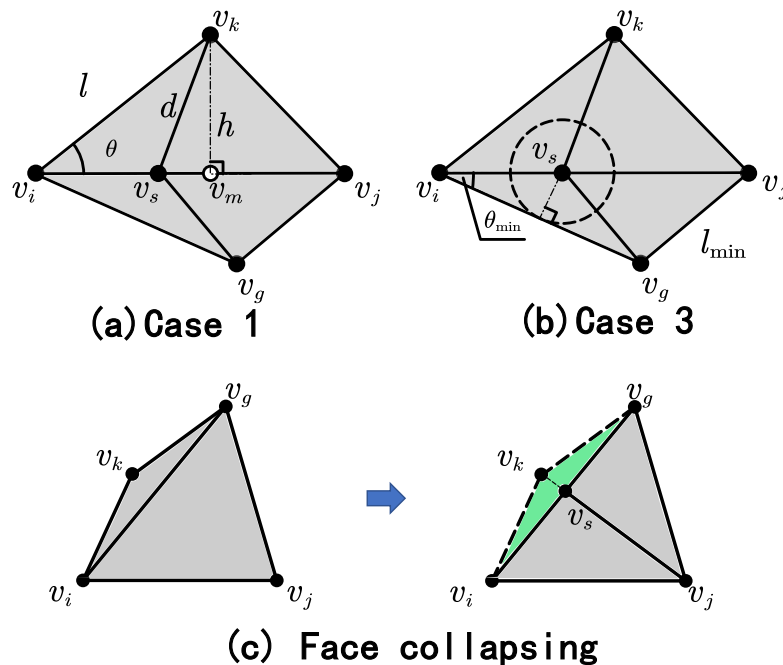


Figure 4. (a) Schematic diagram of Case 1; (b) schematic diagram of Case 3; (c) face collapsing.

A Delaunay mesh construction algorithm is designed and the pseudo code is shown in Algorithm 1. We try to implement the stack using the priority queue approach. Firstly, the control factor δ is initialized. Then, we place all non-local Delaunay edges in a priority queue Q keyed on the sum of the diagonals corresponding to edge e with the largest sum at the top. The algorithm iterative inserts a newly vertex v_s to split the NLD edge from the queue, and updates the information about all elements in $\Omega(v_s)$ such as connectivity and normal vectors. Finally, the edges in the $D(v_s)$ should be determined if they are the NLD edges and all of the new NLD edges will be pushed in the priority queue Q . The algorithm will repeat this procedure until the priority queue Q is empty.

Algorithm 1: Delaunay mesh construction.

```

Input: Arbitray watertight triangle Mesh  $M = (V, E, F)$ 
Output: A Delaunay Mesh  $DM = (V_D, E_D, F_D)$ 
1 Initialization;;
2 First, initialize control factor  $\delta$ ;
3 Second, initialize the priority queue  $Q$  of the all NLD edges.
4 while  $Q$  is not empty do
5   Pop the NLD edge  $e_{ij}$  with the largest diagonal sum from  $Q$ ;
   /* Determine whether triangles are a narrow triangles */
6    $t_{ijk} \leftarrow T(v_i, v_j, v_k)$ ,  $t_{ijg} \leftarrow T(v_i, v_j, v_g)$ ;
7   if  $d_k \leq \delta$  then
8     | collapseTriangle( $t_{ijk}$ );
9   end
10  if  $d_g \leq \delta$  then
11    | collapseTriangle( $t_{ijg}$ );
12  end
   /* Splitting the NLD edges */
13   $e_1 \leftarrow e_{ik}$ ,  $e_2 \leftarrow e_{ig}$ ,  $e_3 \leftarrow e_{jk}$ ,  $e_4 \leftarrow e_{jg}$ ;
14   $r[4] \leftarrow null$ ;
15  for  $m \leftarrow 1$  to 4 do
16    |  $p_m \leftarrow \text{FindCrossPoint}(s_m, e_{ij})$ ;
17    | if  $m == 1 || m == 2$  then
18      | |  $r[m] \leftarrow \text{FindOverlapRange}(v_i, p_m)$ ;
19    | end
20    | if  $m == 3 || m == 4$  then
21      | |  $r[m] \leftarrow \text{FindOverlapRange}(p_m, v_j)$ ;
22    | end
23  end
24   $\text{vert\_insert} \leftarrow \text{FindOverlapPoint}(r[1], r[2], r[3], r[4])$ ;
25  Split edge  $e_{ij}$  at  $\text{vert\_insert}$ ;
   // Update boundary
26  for  $m \leftarrow 1$  to 4 do
27    | if  $e_m$  is a NLD edge then
28      | | Push  $e_m$  into  $S$ .
29    | end
30  end
31 end

```

3.3. The Delaunay Mesh Simplification Algorithm

The complexity of a mesh will increase because of the newly inserted vertices according to the vertex inserting operation. An improved vertex collapsing operation is designed to conquer this drawback. The requirement for the improved vertex collapsing operation is that the local Delaunay property should be preserved in the non-connected domain. We assume that the original mesh is a Delaunay mesh, and the all edges in the set of hollow edges satisfied the local Delaunay property. As shown in Figure 5a, a hollow would be created if a vertex was deleted, and an improved vertex collapsing operation is designed to fill the hollow created by the vertex deletion, whose output is still a Delaunay mesh.

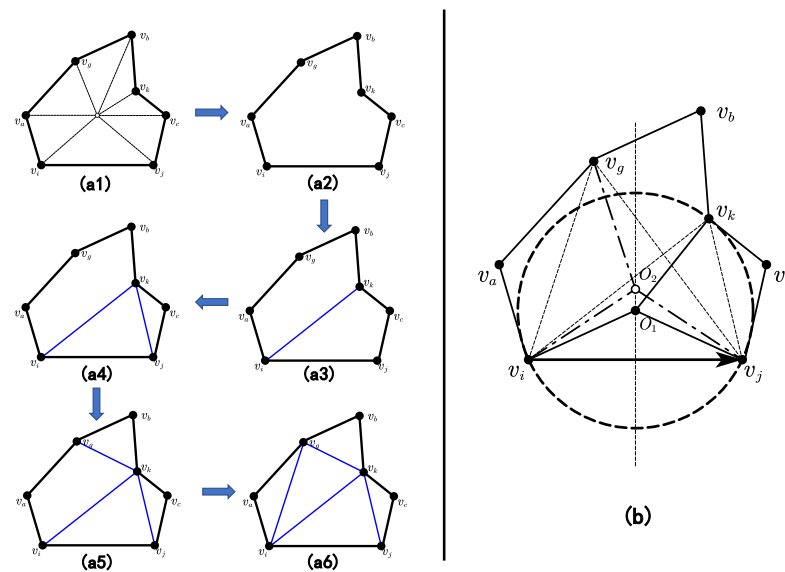


Figure 5. (a) Schematic diagram of hole filling; (b) principle of vertex collapsing.

The principle of the improved operation is shown in Figure 5b. The edge e_{ij} is in the set of hollow-edges, and the set of vertices V is all the vertices that are located in the hollow-edges. We assume that e_{ij} is a directed edge with the certain direction from vertex v_i to vertex v_j . From the principle of the empty geodesic circumcircle property, for any triangle containing edge e_{ij} , there are no other vertices in its circumcircle. Thus, the key of vertices collapsing operation is to find the minimum distance between the circumcenter O of a triangle t and the edge e_{ij} . The specific situation is represented as shown in Figure 5b, where O_1 is the circumcenter of t_{ijk} , and the distance from O_1 to e_{ij} is the shortest; then, the triangle t_{ijk} will satisfy the empty geodesic circumcircle property. Setting the direction of an edge and finding a vertex to the left of e_{ij} can avoid errors in the results because of the concave vertex just like v_k . After all the edges in the set of hollow-edges rebuild the required vertices, the hollow would be patched. The local-Delaunay property of the interior edges, which are inserted after hollow-patching, could be preserved as well.

It is obvious that the improved vertex collapsing operation could be able to simplify a Delaunay mesh. Even if there are NLD edges in the original mesh, this operation can still work on the vertex v_i as long as the edges $e_{jk} \in D(v_i)$ satisfy the local Delaunay property. Therefore, a simplification algorithm based on minimal volume destruction can be designed. The pseudo code is named as the pseudo-code Algorithm 2, where the iterative cost of v_i is set as the volume of $\Omega(v_i)$ (see Figure 6). Firstly, we classify vertex v_i as the non-collapsible and collapsible vertex according to the number of NLD edges in $D(v_i)$, respectively. If there are no NLD edges in $D(v_i)$, the vertex v_i is classified as the collapsible vertex. In this cases, all edges $e_{jk} \in D(v_i)$ satisfy the local Delaunay property, and the improved vertex collapsing operation is worked on v_i , which will not destruct the local Delaunay property of all edges in $D(v_i)$. On the contrary, if there are NLD edges in $D(v_i)$, the improved vertex inserting operation would choose suitable locations on the NLD edge for newly inserted vertices. Then, a hybrid mechanism combined the improved vertex inserting operation and vertex collapsing operation will repeat until the number of vertices in the simplified mesh DM matches the user-defined value.

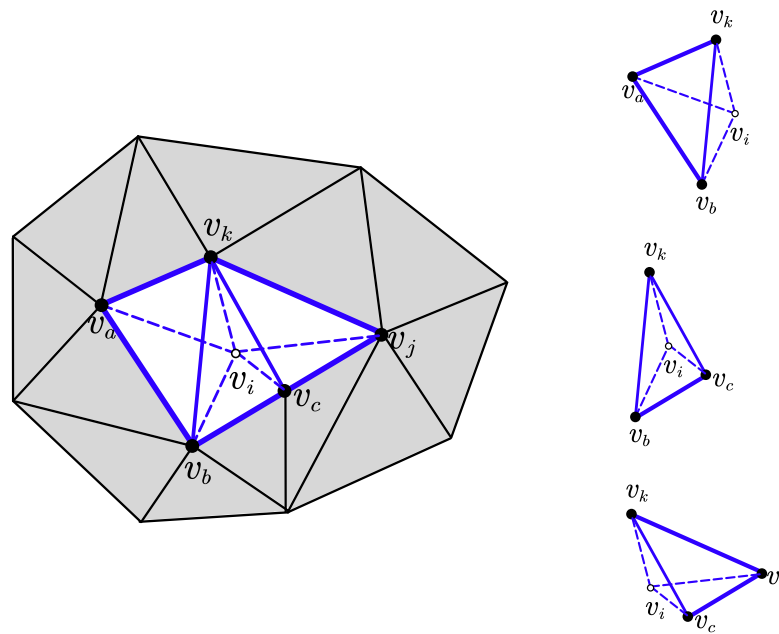


Figure 6. The expression of the sum of volume as an iterative constraint.

Algorithm 2: Delaunay mesh simplification.

Input: Arbitray closed triangle Mesh $M = (V, E, F)$

Output: A simplified Delaunay Mesh $DM = (V_D, E_D, F_D)$ with N vertices

1 **Function** calCost(*vert*):

```

2   | num_of_NLD ← is_NLD( $e \in \Omega(\text{vert})$ ); triangles ← hollowPatching(vert);
3   | volume ← 0;
4   | for t : triangles do
5   |   | volume ← get_volume(t, vert);
6   | end
7   | return volume - num_of_NLD;
```

8 **Initialize**

- the control factor δ ;
- the priority queue Q of the all NLD edges.;
- the cost of each vertex ;

repeat

```

   | /* type_I is a non-collapsible vertex, which has NLD edges in
   |   |  $D(\text{vert})$  */
   | /* type_II is a collapsible vertex, which has no NLD edges in
   |   |  $D(\text{vert})$  */
9   | Divide the vert into type_I and type_II;
10  | if vert  $\in$  type_I then
11  |   | Enhanced_Vert_Inserting(vert);
12  | end
13  | if vert  $\in$  type_II then
14  |   | Enhanced_Vert_Collapsing(vert);
15  | end
16  | Update the cost of vertices for every  $v \in \Omega(\text{vert})$  by using the functon
   |   | calCost(vert);
```

until the number of vertices in the simplified mesh DM matches the user-specified value.;

4. Experimental and Discussion

In this section, experimental results of the construction and simplification algorithm are demonstrated and discussed. All these results are conducted and obtained on a PC with an Intel Core i7-8750H CPU 2.20 GHz and 16.0 GB RAM, and all the codes are implemented in C++.

4.1. Evaluations of the Algorithm Effectiveness

In this paper, the performance of the proposed algorithms is defined as the degree of geometric feature restoration of the original meshes and the accuracy of numerical simulation. Accordingly, two kinds of measures are proposed to evaluate the effectiveness of the algorithms. Firstly, for the degree of geometric feature restoration of the original meshes, the use of visual renderings allows for the visual and intuitive assessment. In addition, volume and surface area are the basic geometric information of a model. We set the geometric feature parameter r as the ratio is a surface-area-to-volume to measure the geometric information of the triangular mesh. Volume, surface area, and their ratio r can be calculated by the following equations, where v_i, v_j, v_k are vertices on a triangle t_{ijk} :

$$\begin{cases} volume = \sum \frac{1}{6}(v_i \times v_j) \cdot v_k, \\ surface_area = \sum \frac{1}{2}(v_j - v_i) \times (v_k - v_i), \\ r = \frac{surface_area}{volume}, \end{cases} \tag{6}$$

Secondly, to demonstrate the effectiveness of numerical simulations, we used a geodesic distance based on the gradient calculation to simulate the thermal diffusion of isotropic simulation [8]. The accuracy of the computation of the scalar function determines the accuracy of the final numerical simulation, where the discrete Laplace operator is defined as a key factor, in the computation of the gradient of the scalar function. In particular, when calculating the gradient of thermal diffusion (see Figure 7), it can be specifically expressed as

$$\Delta u_i = \frac{1}{2A_i} \sum (\cot \alpha_{ij} + \cot \beta_{ij})(u_j - u_i) \tag{7}$$

Here, u is a scalar function defined on the triangle mesh, and A_i is one third of the area of all triangles incident on vertex v_i . The sum is calculated by adding up all neighboring vertices $v_j \in \Omega(v_i)$, where α_{ij}, β_{ij} are the angles opposing to the corresponding edge $e_{ij} \in \Omega(v_i)$.

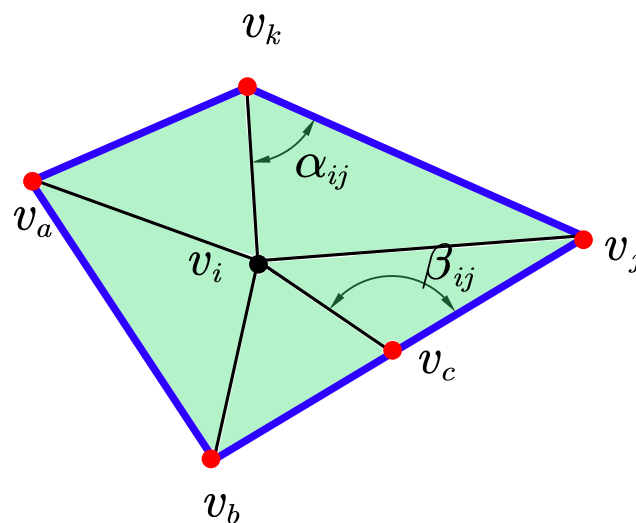


Figure 7. The cot formula for a scalar function on the mesh.

4.2. Performance of the Construction Algorithm

Algorithm 1 is supposed to transform an arbitrary triangular mesh into a Delaunay mesh by inserting a finite number of vertices. For each NLD edge in the original mesh, the inserting operation may change the local Delaunay property of the edge. When there are narrow triangles connected to the NLD edge, it will result in more inserting vertices. As mentioned in Section 3.2, the control factor δ is defined to identify the narrow triangles and the improved face collapsing operation is used to handle this situation. Thus, for the construction algorithm of Delaunay mesh, δ is the most important parameter that determines the space complexity, represented by the number of vertices of the constructed Delaunay mesh. The theoretical lower bound of δ is $l_{min} \cdot \sin(\theta_{min})$.

In this experiment, $\delta = 2 \cdot Average_length \cdot \sin 15^\circ$ was selected as the suitable empirical parameter for the algorithms, which was able to achieve good performance for all the experimental meshes. In Figure 8, a triangle mesh “dinosaur” is used to demonstrate the effects of variable δ . We take $l_{min} \cdot \sin(\theta_{min})$ as the smallest value and $2 \cdot Average_length \cdot \sin 15^\circ$ as the largest value, and select five data in that range to display the relationship between the number of vertices and δ . The rendering results of the smallest and largest values are shown. As shown in Figure 8a, the red curve depicts the relation between the control factor δ and the number of vertices of the constructed Delaunay mesh. With the increase of control factor δ , the judgment criterion of the narrow triangles is amplified. Fewer vertices will be inserted because the face collapsing operation will be used before that. The geometric feature information of the triangle mesh is represented by the parameter r , which is the ratio of surface area to volume of the triangular mesh. As the control factor δ increases, the change of the parameter r remains almost constant. This indicates that the variation of the control factor δ within a reasonable range has little effect on the geometric feature information of the Delaunay mesh. The constructed Delaunay meshes obtained by the two special control factors are rendered as shown in Figure 8b. When the control factor is set as $\delta = l_{min} \cdot \sin(\theta_{min})$, more vertices will be inserted because the narrow triangles generated from the face collapsing operation are too small. If the control factor is set as $\delta = 2 \cdot Average_length \cdot \sin 15^\circ$, the density of vertices in local areas can be effectively improved.

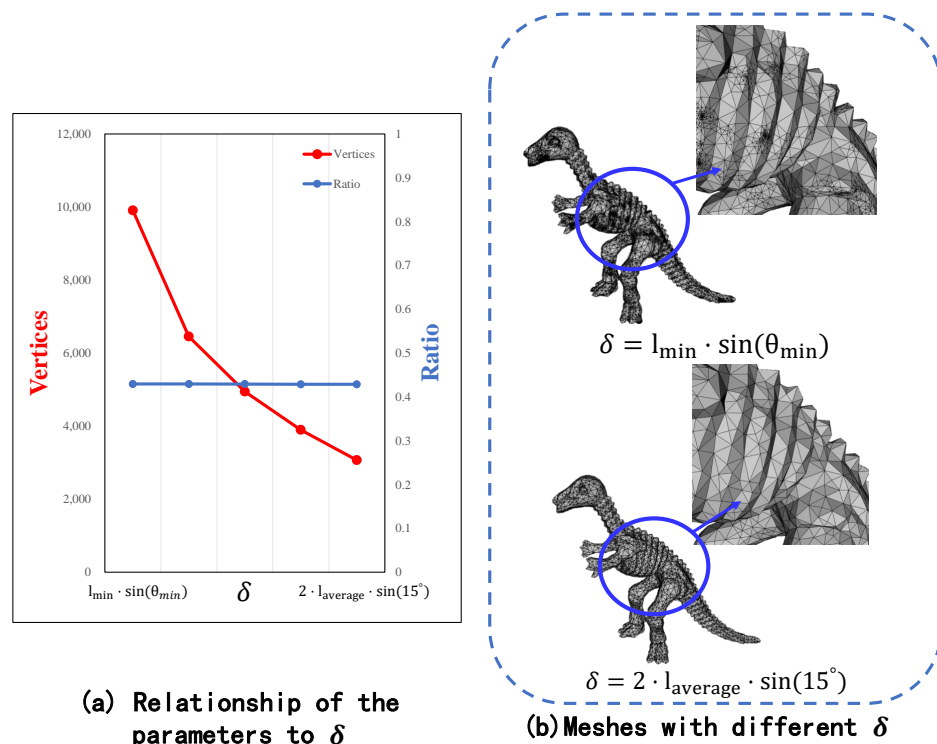


Figure 8. The results of parameter δ selection with the model of “dinosaur”.

The mesh renderings demonstrate the preservation of the geometric feature information. The thermal diffusion renderings are performed to illustrate the calculation effectiveness in numerical simulations. In Figure 9a, the original mesh “refinement” has 510 vertices. Due to the presence of many NLD edges in the original mesh, the results of its thermal diffusion simulation have many errors. For example, in the region A, the isotherms represented by the black line are not consistent with uniform diffusion. In the region B, necessary data information is missing because of too few vertices. The above two conditions show that a non-Delaunay mesh has the lower accuracy in numerical simulation. Figure 9b shows the constructed Delaunay mesh and the thermal diffusion rendering of it. After the processing of Algorithm 1, the Delaunay mesh with 688 vertices is obtained. It can be demonstrated that the constructed Delaunay mesh with 688 vertices can avoid the errors in the thermal diffusion simulation compared to results of the original mesh. In addition, we can find by the geometric feature parameter r that the algorithm proposed in this paper has a better preservation of the geometric feature information of the mesh. In order to demonstrate the performance of the proposed method, the results of the classical area-equalizing triangulation method [4] and the Delaunay mesh construction algorithm [13] are performed with the same model and shown in Figure 9c and Figure 9d, respectively. The selected area-equalizing triangulation method is a basic algorithm for model meshing of finite element method (FEM), and Liu’s method [13] is a typical representation for the construction of the Delaunay mesh. The number of vertices using their methods are 7003 and 7907, respectively. It is obvious that the construction algorithm proposed in this paper has better performance in the aspects of simulation accuracy with less vertex numbers.

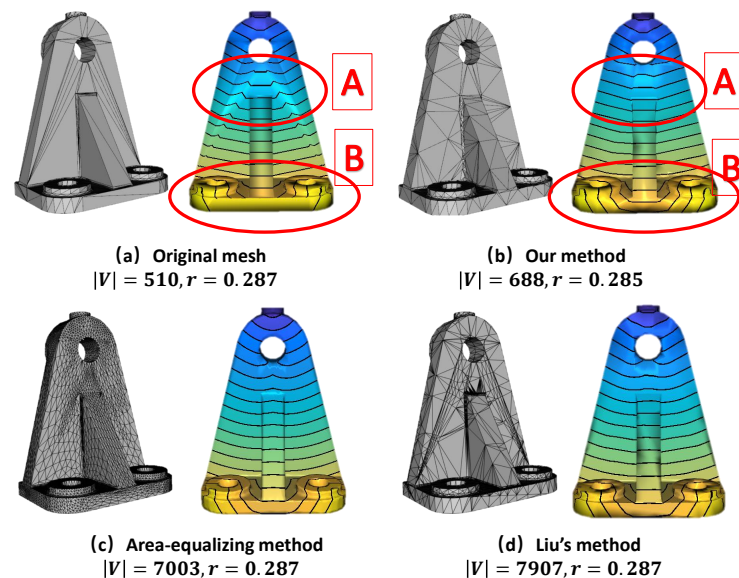


Figure 9. The comparison of the construction algorithms with numerical simulation results, where regions A and B are significant changes.

4.3. Performance of the Simplification Algorithm

After the construction algorithm of Delaunay mesh via Algorithm 1, any triangle mesh can be transformed to the Delaunay mesh. However, the inserted vertices may lead to the increase of the mesh complexity and the data storage. Algorithm 2 provides a Delaunay mesh simplification algorithm that can maintain the local Delaunay property of the Delaunay mesh and transform the non-Delaunay mesh to the Delaunay mesh.

A model of “bumpy” is used to show the effect of the simplification algorithm on the preservation of the geometric information. The original mesh with 1250 vertices is the Delaunay mesh, and the improved vertex collapsing operation is used to simplify it. As shown in Figure 10, the un-simplified mesh and the simplified meshes with 80% vertices, 40% vertices, and 20% vertices are displayed, respectively. Both the visualization and the

ratio r , surface-area-to-volume, demonstrate that there is better preservation of geometric information as the number of vertices of the simplified mesh is reduced.

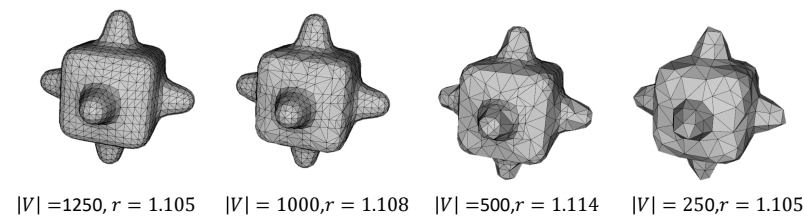
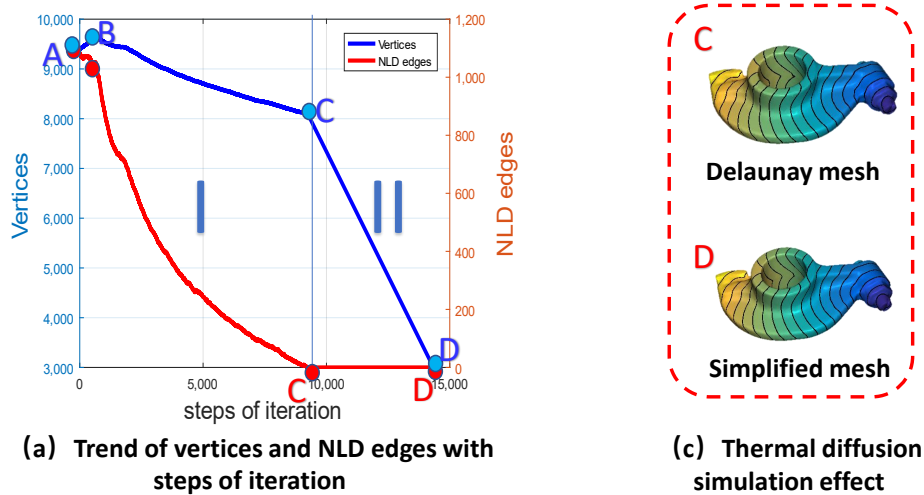


Figure 10. The simplification results for a Delaunay mesh using the model of “bumpy”.

Similarly, the simplification algorithm also works for non-Delaunay mesh. The variation trends of the number of vertices and the number of NLD edges with the iteration steps are shown in Figure 11. The blue curve indicates the trend of vertices versus iteration steps, and the red curve represents the trend of NLD edges versus iteration steps. The input mesh of Algorithm 2 is a non-Delaunay mesh with 9397 vertices and 1082 NLD edges, which are illustrated as A. The processing of algorithm can be divided into two parts. According to the simplification algorithm described in Section 3.3, for a non-Delaunay mesh, it firstly classifies all vertices into two types. For all of the non-collapsible vertices, it can be found that the number of the mesh vertices increased with the number of iterations of the algorithm in region-I of Figure 11a. Subsequently, when all the vertices in the mesh are collapsible vertices, which is illustrated as B, the number of vertices start to decrease. As the iterative step proceeds, the Delaunay mesh based on minimal volume destruction can be obtained, which is illustrated as C. In the region-II of Figure 11a, the NLD edges disappear, and the simplification algorithm is executed repeatedly until the user defined value is satisfied.



(a) Trend of vertices and NLD edges with steps of iteration

(c) Thermal diffusion simulation effect

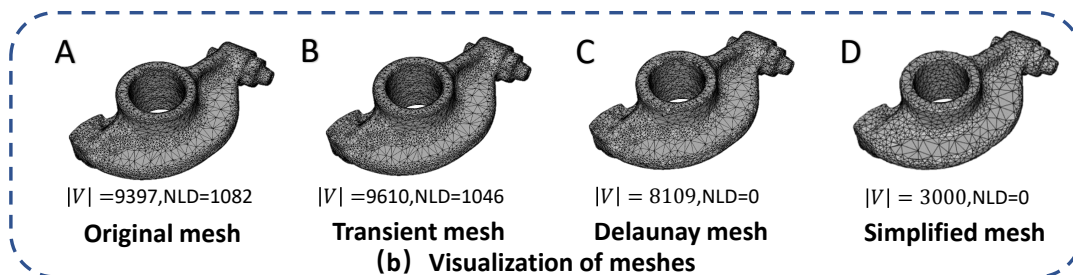


Figure 11. The results after the hybrid construction and simplification algorithm with the model of “rocker”.

The visualization of the meshes of A, B, C and D is shown in Figure 11b, respectively. As mentioned above, A is the original mesh with 9397 vertices and 1082 NLD-edges, B is a non-Delaunay meshes processed by the vertices inserting operation, C is the Delaunay mesh obtained by the Algorithm 2 with 8109 vertices, and D is a simplified mesh with 3000 vertices. Especially, the thermal diffusion simulation results for C and D are also shown in Figure 11c, respectively. As the number of vertices reduce, the geometric features of the original mesh can still be effectively displayed. These results demonstrate the effectiveness of the simplification algorithm in preserving geometric information. In addition, the thermal diffusion rendering of the two simplified meshes also show that the constructed Delaunay meshes can also achieve a good performance in numerical simulation. For the case of D, it is possible to obtain similar numerical simulation results in the case of C with fewer vertices. It will enable the simplification of both the Delaunay mesh and the non-Delaunay mesh without destroying the performance of the thermal diffusion simulation.

In order to further illustrate the generality of Algorithm 2, three more experiments are conducted and shown in Figure 12. The processed models are “elephant”, “horse”, and “head” of non-Delaunay mesh, respectively. After the processing of Algorithm 2, all the meshes can be transformed into the Delaunay meshes and the number of vertices is reduced. To demonstrate the advantage of the simplification algorithm, the methods of the classical QEM [3] and the Delaunay mesh simplification [13] were conducted and illustrated as the contrast. All these two methods and the proposed method in this paper are simplified to the same number of vertices. The QEM method [3] is considered as a general control experiment for the simplification algorithm because of its good performance in the aspect of geometric feature preservation of the model. However, it is noteworthy that the results obtained with this method do not have the local and global properties of Delaunay mesh. There are still many narrow triangles among the simplified meshes, which are easy to be visually detected by its surface meshes. Although Liu’s method [13] can preserve the local and global properties of the Delaunay meshes, it is visually noticeable that the meshes obtained by our method are more uniform with the same level of simplification.

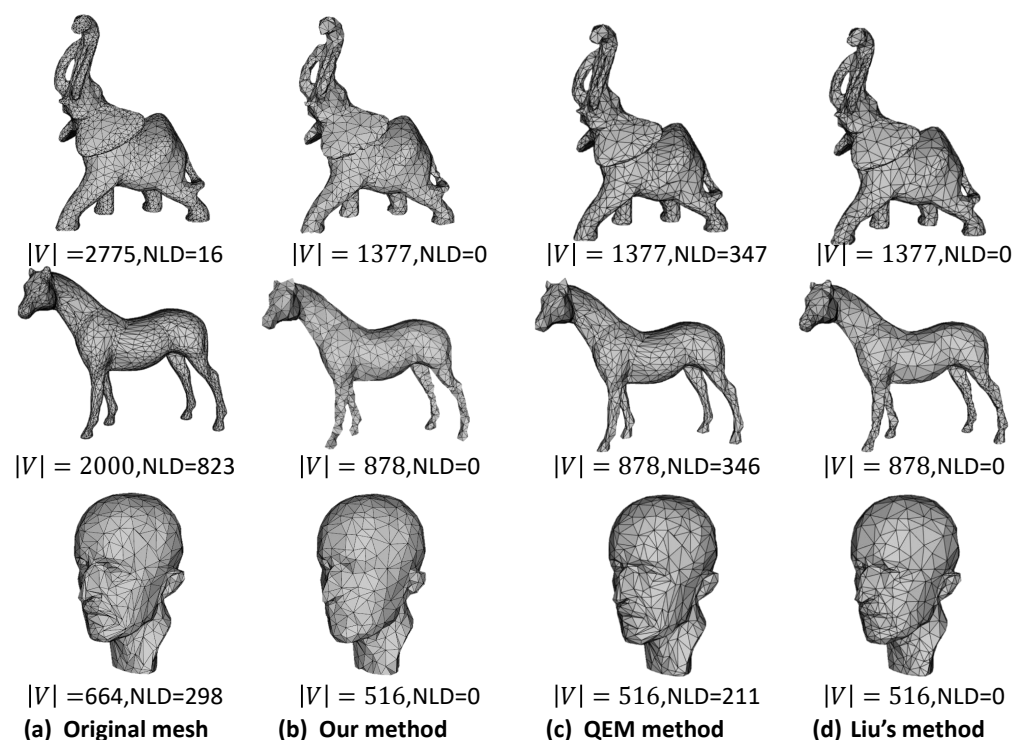


Figure 12. The comparison of the processed effects of the Delaunay mesh, (a) the original mesh, (b) our method, (c) QEM method, and (d) Liu’s method.

To better illustrate the preservation level of the geometric features after the simplification, the ratios surface-area-to-volume r of these four models are calculated and listed in Table 1. The algorithm proposed in this paper is based on the principle of minimal volume destruction. It can be seen that the vertices of mesh gradually decreases when the quantity of the meshes is improved. As the illustration factor of geometric features, the quantification criterions r of all these methods differ slightly. The values of Δr are defined as the geometric feature variations of the meshes compared to the original mesh. The small Δr value means that the changes to the geometric feature of the simplified mesh is little. The results show that the geometric features of all these meshes can be preserved after the simplification algorithm. In particular, for the original mesh with a large number of vertices, such as “rocker”, “elephant”, and “horse”, the simplified mesh of the proposed method has smaller Δr compared to the simplified meshes of other methods. However, since the proposed method is based on minimal volume destruction, when the original model of “head” with fewer vertices is simplified, the vertex collapsing may result in the undesired volume destruction. Therefore, the proposed method in this paper is not applicable to process meshes with few vertices. All in all, it is demonstrated that the method proposed in this paper can protect the local and global properties of the Delaunay mesh and preserve the geometric features of the original mesh in most cases.

Table 1. The geometric feature information of the meshes.

Meshes	Vertices	Volume	Surface Area	r	Δr
rocker_original	9397	0.0425	1.2965	30.4965	0
rocker_our	3000	0.0423	1.28926	30.4283	0.00223
rocker_qem [3]	3000	0.0424	1.30145	30.6283	0.00432
rocker_liu [13]	3000	0.0423	1.28805	30.4141	0.00270
elephant_original	2775	0.046201	1.24496	26.9465	0
elephant_our	1377	0.045656	1.22927	26.925	0.00079
elephant_qem [3]	1377	0.04609	1.24725	27.061	0.00424
elephant_liu [13]	1377	0.045757	1.24068	27.11	0.00606
horse_original	2000	0.157444	2.49807	15.8664	0
horse_our	878	0.153055	2.44337	15.964	0.00615
horse_qem [3]	878	0.156998	2.50834	15.9769	0.00696
horse_liu [13]	878	0.153371	2.46727	16.087	0.01390
head_original	664	5.62153	0.183606	0.032661	0
head_our	516	5.60071	0.181661	0.032435	0.00691
head_qem [3]	516	5.61976	0.183612	0.032673	0.00035
head_liu [13]	516	5.5796	0.18225	0.032664	0.00007

5. Conclusions

In this paper, the Delaunay construction and simplification algorithms were proposed to improve the model complexity and simulating accuracy. Firstly, we proposed two improved remeshing operations of the vertex inserting and the vertex collapsing. Then, the improved algorithms for the construction and simplification of Delaunay mesh based on minimal volume destruction were presented. The Delaunay construction algorithm can transform the arbitrary mesh into the Delaunay mesh by inserting finite numbers of vertices. The simplification algorithm can classify the vertices into collapsible and non-collapsible vertices, which will be performed differently according to the types of vertices. Moreover, the renderings of triangular meshes demonstrated the accuracy of the proposed algorithms. Finally, the experimental results showed that the algorithm can improve the quality of the arbitrary meshes. It is also noteworthy that, when the original model with fewer vertices is simplified, the vertex collapsing may result in the undesired volume destruction. The similar limitation occurs when the meshes with a large number of vertices were simplified to a high level, for example to 20% of the original vertices. Then, the cost of the algorithm

will increase significantly and the geometric feature information of the simplified mesh may degenerate. To minimize the destruction to the original mesh, the control factor δ was designed to judge the narrow triangle. At present, the value of the control factor δ should be based on the user defined values. We will try to realize the automatic and real-time optimization of the control factor in the future study. In our future work, the proposed algorithms will be applied to 3D model processing for stereo-lithography additive manufacturing and experiments will be conducted to evaluate the performance.

Author Contributions: The paper's initial idea was proposed by Y.H., T.W., B.L. and G.W. The conceptualization is by Y.H. and T.W.; the software work and analysis were performed by Y.H., H.L., X.L. and Y.Z.; Supervision by T.W. and G.W. The original draft preparation was by Y.H. and T.W. and the review and editing were performed by Y.H. and T.W. All authors have read and agreed to the published version of the manuscript.

Funding: The authors gratefully acknowledge the financial support of the National Natural Science Foundation of China (Grant No. 52005479), the Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences (Project No. CSU-JJKT-2020-6) and the Scientific Instrument Developing Project of the Chinese Academy of Sciences (Grant No. YJKYYQ20200068).

Conflicts of Interest: We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service and company that could be construed as influencing the position presented in, or the review of, the manuscript entitled.

References

1. Alliez, P.; Ucelli, G.; Gotsman, C.; Attene, M. Recent Advances in Remeshing of Surfaces. In *Shape Analysis and Structuring*; De Floriani, L., Spagnuolo, M., Eds.; Publishing House: Berlin/Heidelberg, Germany, 2008; pp. 53–82.
2. Hoopé, H.; Derose, T.; Duchamp, T. Mesh optimization. In Proceedings of the Conference on Computer Graphics and Interactive Techniques, Anaheim, CA, USA, 2–6 August 1993.
3. Garland, M.; Heckbert, P. Surface Simplification Using Quadric Error Metrics. In Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, Los Angeles, CA, USA, 3–8 August 1997.
4. Botsch, M.; Kobbelt, L. *A Remeshing Approach to Multiresolution Modeling*; Association for Computing Machinery: New York, NY, USA, 2004; pp. 189–196.
5. Wang, Y.Q.; Yan, D.M.; Liu, X.H.; Tang, C.C.; Guo, J.W.; Zhang, X.P.; Wonka, P. Isotropic Surface Remeshing without Large and Small Angles. *IEEE Trans. Vis. Comput. Graph.* **2019**, *25*, 2430–2442. [[CrossRef](#)] [[PubMed](#)]
6. Wang, Q.; Gao, B.; Wu, H. Triangular mesh generation on free-form surfaces based on bubble dynamics simulation. *Eng. Comput.* **2019**, *36*, 646–663. [[CrossRef](#)]
7. Khan, D.; Plopski, A.; Fujimoto, Y.; Kanbara, M.; Cheng, Z.; Kato, H. Valence optimization and angle improvement for molecular surface remeshing. *Vis. Comput.* **2020**, *36*, 2355–2368. [[CrossRef](#)]
8. Crane, K.; Weischedel, C.; Wardetzky, M. Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Trans. Graph.* **2013**, *32*, 152–162. [[CrossRef](#)]
9. Bobenko, A.I.; Springborn, B.A. A Discrete Laplace—Beltrami Operator for Simplicial Surfaces. *Discrete Comput. Geom.* **2007**, *38*, 740–756. [[CrossRef](#)]
10. Ye, Z.; Yi, R.; Gong, W.; He, Y.; Liu, Y.-J. Dirichlet energy of Delaunay meshes and intrinsic Delaunay triangulations. *Comput. Aided Des.* **2020**, *126*, 102851. [[CrossRef](#)]
11. Dyer, R.; Zhang, H.; Möller, T. Voronoi-Delaunay Duality and Delaunay Meshes. In Proceedings of the 2007 ACM Symposium on Solid and Physical Modeling, Beijing, China, 4–6 June 2007.
12. Dyer, R.; Zhang, H.; Möller, T. Delaunay Mesh Construction. In Proceedings of the Fifth Eurographics Symposium on Geometry Processing, Barcelona, Spain, 4–6 July 2007.
13. Liu, Y.; Xu, C.; Fan, D.; He, Y. Efficient construction and simplification of Delaunay meshes. *ACM Trans. Graph.* **2015**, *34*, 174–186. [[CrossRef](#)]
14. Yi, R.; Liu, Y.; He, Y. Delaunay mesh simplification with differential evolution. *ACM Trans. Graph.* **2018**, *37*, 263–274. [[CrossRef](#)]
15. Rivin, I. Euclidean Structures on Simplicial Surfaces and Hyperbolic Volume. *Ann. Math.* **1994**, *139*, 553–580. [[CrossRef](#)]
16. Leibon, G.; Letscher, D. Delaunay Triangulations and Voronoi Diagrams for Riemannian Manifolds. In Proceedings of the Annual Symposium on Computational Geometry, Hong Kong, China, 12–14 June 2000.
17. Chen, Z.; Wang, W.; Levy, B.; Liu, L.; Sun, F. Revisiting Optimal Delaunay Triangulation for 3D Graded Mesh Generation. *SIAM J. Sci. Comput.* **2014**, *36*, 930–945. [[CrossRef](#)]
18. Aurenhammer, F. Voronoi diagrams—A survey of a fundamental geometric data structure. *ACM Comput. Surv.* **1991**, *23*, 345–405. [[CrossRef](#)]

19. Guibas, L.; Stolfi, J. Primitives for the manipulation of general subdivisions and the computation of Voronoi. *ACM Trans. Graph.* **1985**, *4*, 74–123. [[CrossRef](#)]
20. Liu, Y.; Fan, D.; Xu, C.; He, Y. Constructing Intrinsic Delaunay Triangulations from the Dual of Geodesic Voronoi Diagrams. *ACM Trans. Graph.* **2017**, *36*, 1–15. [[CrossRef](#)]
21. Liu, Y.; Wang, W.; Lévy, B.; Sun, F.; Yan, D.; Lu, L.; Yang, C. On centroidal voronoi tessellation—Energy smoothness and fast computation. *ACM Trans. Graph.* **2009**, *28*, 101–117. [[CrossRef](#)]
22. Wang, X.; Ying, X.; Liu, Y.; Xin, S.; Wang, W.; Gu, X.; Mueller-Wittig, W.; He, Y. Intrinsic computation of centroidal Voronoi tessellation (CVT) on meshes. *Comput. Aided Des.* **2015**, *58*, 51–61. [[CrossRef](#)]
23. Liu, Y.; Xu, C.; Yi, R.; Fan, D.; He, Y. Manifold differential evolution (MDE): A global optimization method for geodesic centroidal voronoi tessellations on meshes. *ACM Trans. Graph.* **2016**, *35*, 243–252. [[CrossRef](#)]