RESEARCH ARTICLE

Quality improvement of surface triangular mesh using a modified Laplacian smoothing approach avoiding intersection

Tiantian Liu[1,2], **Minxin Chen**[1]*, **Yu Song**[3], **Hongliang Li**[3], **Benzhuo Lu**[2]*

1 Department of Mathematics, Soochow University, Suzhou 215006, China, **2** State Key Laboratory of Scientific and Engineering Computing, National Center for Mathematics and Interdisciplinary Sciences, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China, **3** Microsystem & Terahertz Research Center, China Academy of Engineering Physics, Chengdu, China

* chenminxin@suda.edu.cn (MC); bzlu@lsec.cc.ac.cn (BL)

OPEN ACCESS

Citation: Liu T, Chen M, Song Y, Li H, Lu B (2017) Quality improvement of surface triangular mesh using a modified Laplacian smoothing approach avoiding intersection. PLoS ONE 12(9): e0184206. https://doi.org/10.1371/journal.pone.0184206

Editor: Yongtang Shi, Nankai University, CHINA

Received: March 20, 2017

Accepted: August 16, 2017

Published: September 8, 2017

Data Availability Statement: Data are available from https://doi.org/10.6084/m9.figshare.5346169.v1.

Funding: This work was funded by the Science Challenge Project (No. TZ2016003), China NSF (NSFC 91530102, NSFC 21573274, NSFC 11301368, NSFC 11404300) and NSF of Jiangsu Province (BK20130278).

Competing interests: The authors have declared that no competing interests exist.

Abstract

We present a systematic procedure to improve the qualities of triangular molecular surface meshes and at the same time preserve the manifoldness. The procedure utilizes an algorithm to remove redundant points having three or four valences and another algorithm to smooth the mesh using a modified version of Laplacian method without causing intersecting triangles. This approach can be effectively applied to any manifold surface meshes with arbitrary complex geometry. In this paper, the tested meshes are biomolecular surface meshes exhibiting typically highly irregular geometry. The results show that the qualities of the surface meshes are greatly improved and the manifoldness of the surface meshes are preserved. Compared with the original meshes, these improved molecular surface meshes can be directly applied to boundary element simulations and generation of body-fitted volume meshes using Tetgen. The procedure has been incorporated into our triangular molecular surface mesh generator, TMSmesh 2.0. It can be also used as a standalone program and works together with any other surface triangular mesh generator to obtain qualified manifold mesh. The package is downloadable at https://doi.org/10.6084/m9.figshare.5346169.v1 and can be run online at http://www.xyzgate.com.

Introduction

Surface mesh generation arises in many applications, such as numerical simulation, computer visualization and geometry processing. Most computational applications involve triangulation of a complex surface geometry, especially in computational biology. Molecular surface plays an important role in computational biology, such as protein folding, structure prediction, docking and implicit solvent modeling. Recent developments in realistic mathematical modeling and numerical simulation of biomolecular systems raise new demands for qualified, stable, and efficient surface meshing, especially in implicit-solvent modeling [1]. Many triangulated meshes are generated by scanning devices or by isosurfacing implicit representations. However, it is not easy to generate high-quality mesh for complex surface geometry in such processes, especially if automated. Low-quality triangular meshes can undermine the order of accuracy or even cause non-convergence in numerical computations. In this case, it is desirable to build high-quality meshes from those low-quality meshes before performing any numerical simulation. [2]

Mesh quality refers to the faithfulness, manifoldness and uniformness. Faithfulness is measured by how accurately the surface mesh preserves the original geometry and topology, such as surface area, volume and curvature of the referenced surface. Faithfulness related to the accuracy of the numerical simulation and geometry processing. Manifoldness of a surface mesh means that each point on the surface has a neighborhood which is homeomorphic to a disk in a real plane. Non-manifold surface mesh brings difficulties in generating corresponding surface conforming volume mesh that is an essential prerequisite for successful finite element method computations. One of the typical non-manifold errors in surface meshes is intersecting triangles. It is an important issue that preserving the manifoldness of the original surface meshes in the process of improvements of mesh qualities. Uniformness includes the triangle shape, regularity, complexity and so on. The quality of triangles in surface mesh is crucial for robustness of boundary element method (BEM) and finite element method (FEM) computations. In some numerical analysis, such as finite element analysis, regular or qualified mesh is always required. The mesh size is related to the complexity of the mesh. Generally, more sampling points capture more surface details. But more sampling points increase the complexity of the mesh, which increases the difficulty of numerical simulation. Therefore, meshes can be improved with respect to any number of quality metrics including shape, size, manifoldness, solution error or combinations of these. The improvement process is called remeshing. [3] There are two fundamental remeshing approaches, parameterization techniques [4–10] and mesh adaptation strategies [11–15]. In the context of the parameterization-based remeshing, the initial surface is parameterized onto a surface in 2D, the 2D surface is meshed by the standard mesh generation method and the new mesh is projected back to the original surface. The existing parameterization method is classified to linear method [16–21], non-linear method [22–24] and hybrid method [25, 26]. The parameterization-based methods can work even for coarse resolution remeshing, but the computation is expensive and the result is sensitive to the specific parameterization. The mesh adaptation method treats the surface as a set of points in space and operate directly on the surface. The strategies use local mesh modifications to improve the quality of the input surface mesh or adapt the mesh to a given mesh size criterion. Typical techniques for local mesh modifications combine vertex smoothing, vertex insertion/deletion, vertex shift, edge split, edge flip, edge collapse and so on. [10, 27–30]

Recently, we have developed a method, TMSmesh [31, 32] and its improved version, TMSmesh 2.0 [33], to generate molecular surface mesh for large biomolecules. TMSmesh 2.0 succeeds to generate manifold surface meshes for biomolecules comprised of more than one million atoms efficiently. The generated surface mesh preserves the original molecular surface features and properties (topology, surface area, enclosed volume, local curvature and so on) [34]. However, the mesh quality in the aspects of uniformness and smoothness are not guaranteed. There are some redundant points and some singular triangles in the meshes generated by TMSmesh 2.0. In boundary element and finite element simulation for some large biomolecules, the uniformness of the surface mesh is required to be improved to make the numerical simulation more robust and accurate. The faithfulness of the surface mesh to the original geometric features are critical in getting reliable simulation results from BEM and FEM computation in continuum molecular simulations. The parameterization-based methods contain the process of resampling, so these automated operation can not guarantee the fidelity of improved meshes to the original surface. And the complex geometry and topology of the large biomolecules will increase the difficulty of parameterization dramatically. There are some packages using mesh adaption strategies to improve mesh qualities, but few of them can preserve manifold meshes for complex surfaces. Therefore, in this work, we develop a package named SMOPT to improve the mesh quality directly on the surface and perform a series of local modifications on the mesh. The qualities of the new resulted meshes are improved greatly over the original surface meshes and avoid causing non-manifold errors, such as intersecting triangles. The program SMOPT has been incorporated into TMSmesh 2.0. It can be also used as a standalone program and works together with any other surface triangular mesh generator to obtain qualified manifold mesh.

In the next Section, the method used to improve the surface mesh quality is introduced. Some examples, analysis and applications are presented in the results section. The final section, Conclusion, gives some concluding remarks.

Materials and methods

In this section, we describe the mesh generator TMSmesh 2.0 and the improvement procedure SMOPT to improve the quality of the triangular mesh generated by TMSmesh 2.0.

0.1 TMSmesh 2.0

TMSmesh 2.0 is an algorithm for manifold triangular meshing of Gaussian molecular surfaces. [31–33] The Gaussian surface is defined as a level set of the summation of Gaussian kernel functions:

$$\{\vec{x} \in R^3, \phi(\vec{x}) = c\}, \tag{1}$$

where

$$\phi(\vec{x}) = \sum_{i=1}^{N} e^{-d(\|\vec{x}-\vec{x}_i\|^2 - r_i^2)}, \tag{2}$$

$\vec{x}_i$ and r_i are the location and radius of the ith atom, N is the number of atoms in the molecule. d is the decay rate of the Gaussian kernel. When d decreases, the molecules will show more geometrical details. c is the isovalue and it controls the volume enclosed by the Gaussian surface [34].

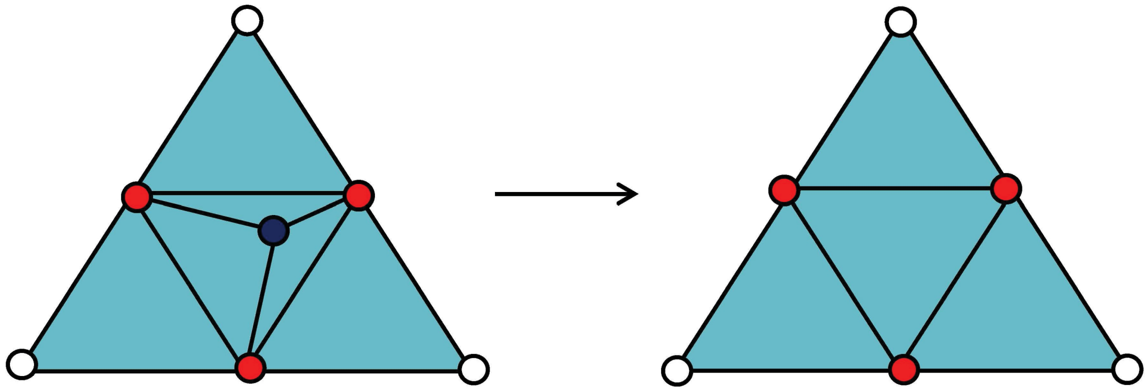


Fig 1. The black point is considered as the redundant point when it has a valence of three and is almost on the plane formed by its 1-ring neighborhoods in red color.

https://doi.org/10.1371/journal.pone.0184206.g001

The algorithm of TMSmesh 2.0 contains two stages, the first stage is an adaptive estimation and division process. The Gaussian surface is approximated by piecewise trilinear surface within controllable error. The second stage is to partition each piece of trilinear surface into single-valued patches along x, y, z directions by tracing along the fold curves. Then each single-valued patch is triangulated by the ear clipping algorithm. TMSmesh 2.0 succeeds to generate surface meshes for biomolecules comprised of more than one million atoms, and all the generated meshes are manifold mesh preserving the original detailed geometry of molecular surface. The sampled points in TMSmesh 2.0 are distributed according to the curvature and geometric features of molecular surface adaptively. In the regions with dramatic changing shapes, the density of the sampled points is high. It may result in some very close points in the

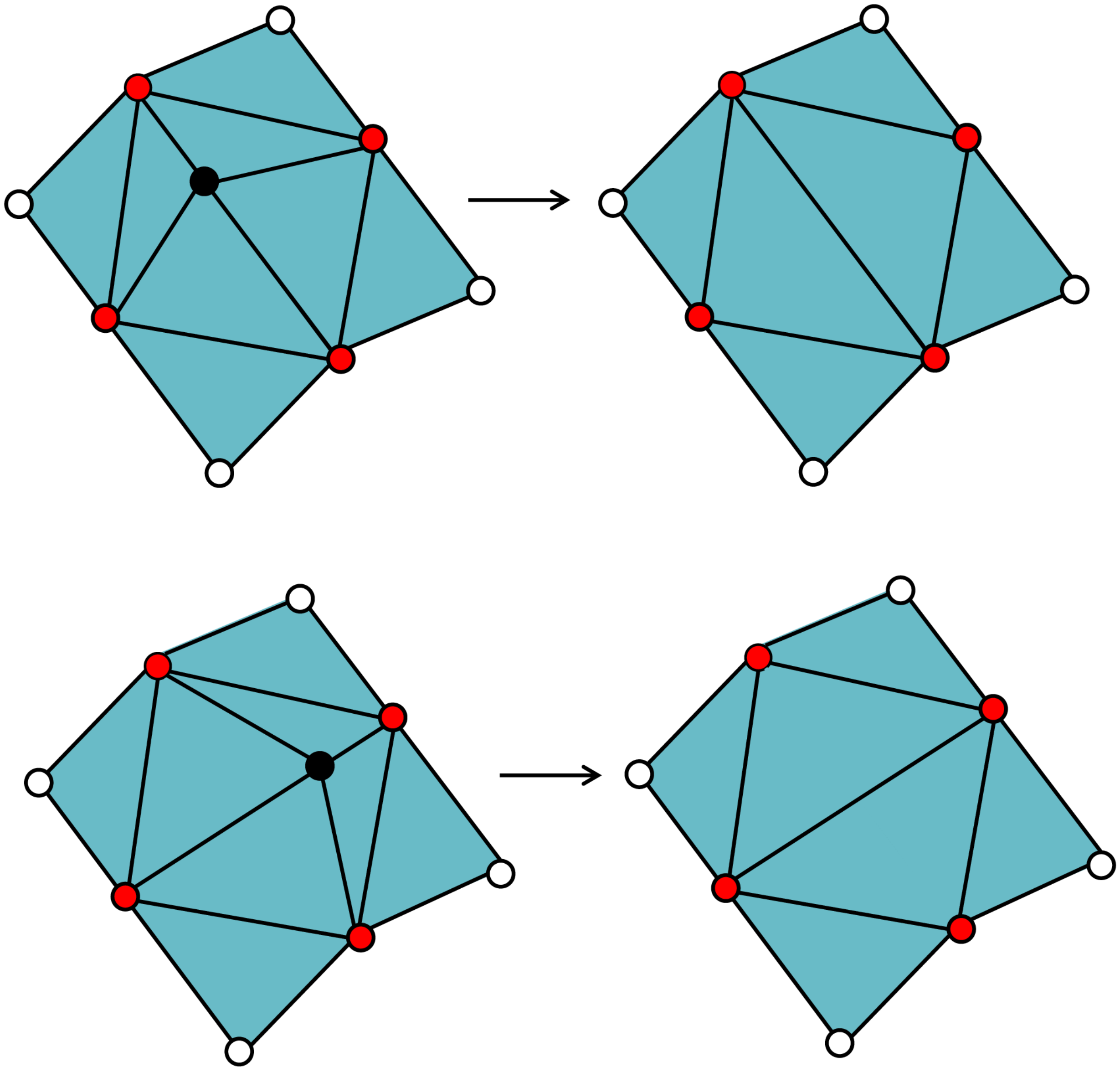


Fig 2. The redundant point whose valence is four. The black point is redundant point. The red points are 1-ring neighborhoods of the black point. The figure shows two cases that the black point is near two different diagonals.

https://doi.org/10.1371/journal.pone.0184206.g002

mesh, which will bring difficulties to generate body-fitted volume mesh. And the very close points may cause singularity in numerical simulations for continuum modeling of biomolecules. Therefore, some methods are required to delete the redundant points and improve the mesh quality without changing the geometric features of molecules.

0.2 Algorithm to delete redundant points

The redundant points are chosen from the vertices whose valence is three or four. Here, valence is the number of the adjacent vertices connected by edges.

Fig 1 shows the redundant condition that the valence of the point is three. When the black point is or almost on the plane formed by its three red connected neighboured points, the black point is considered as a redundant one. In this situation, the black point should be deleted. Meanwhile, the elements and the edges whose vertices contain the black point should also be deleted. Fig 2 shows the case of redundant vertex with a valence of four. When the black point is almost on one of two diagonals of the polygon formed by the four red connected neighboured points, the black point is considered as a redundant one. In this case, the black vertex, the elements and the edges whose vertices contain the black vertex should be deleted, and the diagonal on which the black vertex located should be added as a new edge. Then we can get two new triangles to replace the original four triangles containing the redundant vertex. Algorithm 1 shows the whole process to delete the redundant vertices.

The valence three and four cases are easy to handle. When the valence is three, we can delete the point directly. When the valance is four, we delete the point and re-triangulate the resulted polygon with four vertices. For this two simple cases, the deletion and re-triangulate processes will not lead to intersections, because these operations do not add any new planes. But for the cases with higher number of valence, after deleting the redundant vertex, we should re-triangulate the resulted polygon composed of the neighborhood triangles of the deleted vertex. It is hard to guarantee no intersection, especially when this polygon is not flat. Therefore, in these cases, we prefer to change the position of the vertices with more than four valence rather than delete.

Algorithm 1 Delete the redundant points.

```
Input: vertices in the mesh and the connection relation of each vertex.
  $i = 1$
  $err = 1e-4$
  while $i \leq$ num of vertices do
    if valence of vertex $i$ is 3 then
      $n_1$, $n_2$, $n_3$ are the points on the neighbour loop of vertex $i$
      $P$ is the plane formed by $n_1$, $n_2$, $n_3$
      if distance ($i$, $P$) < $err$ then
        delete vertex $i$
        delete the triangles whose vertices contain $i$
        insert the new triangle formed by $n_1$, $n_2$, $n_3$
      end if
    end if valence of vertex $i$ is 4 then
      $n_1$, $n_2$, $n_3$, $n_4$ are the points on the neighbour loop of vertex $i$
      $n_1n_3$ and $n_2n_4$ are the diagonals of the neighbour loop
      if distance ($i$, $n_1n_3$) < $err$ then
        delete vertex $i$
        delete the triangles whose vertices contain $i$
        insert two new triangles formed by $n_1$, $n_3$, $n_4$ and $n_1$, $n_2$, $n_3$
      else if distance ($i$, $n_2n_4$) < $err$ then
        delete vertex $i$
        delete the triangles whose vertices contain $i$
        insert two new triangles formed by $n_1$, $n_2$, $n_4$ and $n_2$, $n_3$, $n_4$
      end if
    end if
  end while
```

0.3 Mesh smoothing

An inherent problem of the automated triangulated surface mesh generation is that the resulted polygonal surface appears faceted and may contains singular triangles with very small angles. The original Gaussian surface of the biomolecule is smooth. But the generated mesh can make the approximated surface rough. And the singular triangles may cause difficulties in BEM computation and corresponding body-fitted volume mesh generation. Therefore, smoothing method is necessary for improving the qualities of surface meshes.

Laplacian smoothing is an algorithm to smooth a triangular surface mesh. [35, 36] The Laplacian smoothing technique changes the position of nodes without modifying the topology of the mesh. In the process of Laplacian smoothing, each vertex is move to a new position defined by taking the average position of its neighbors:

$$p_i = \frac{1}{N_i}\sum_{j=1}^{N_i} q_j \tag{3}$$

where N_i is the number of the adjacent vertices of the ith node, q_j is the position of the jth adjacent vertex. To control the rate of smoothing, the current position q_i is also included in the calculation of the new position p_i:

$$p_i = (1-\beta)q_i + \frac{\beta}{N_i}\sum_{j=1}^{N_i} q_j \tag{4}$$

where β is the parameter to control the rate of smoothing. Usually, β takes a value between [0, 1]. However, if the geometrical characters are very complex in a region, the Laplacian smoothing may cause self-intersection. To ensure manifoldness, we proposed the modified version of Laplacian smoothing as follows. For each vertex, initially, β is chosen to be 1 and the vertex is moved to a new position through Eq (4). Then we check whether this movement causes intersection near this vertex. And if it does result in intersecting triangles, redo this movement by Eq (4) using half β. This process is repeated until the new position of current vertex does not bring intersecting triangles. And in each time of checking intersecting triangles, we only need to check whether there is an edge intersecting a triangle in the vicinity of the current moved vertex.

The key to detect self-intersection is to decide whether an edge is intersecting with a triangle. Fig 3 shows the case that an edge intersects with a triangle. In Fig 3, $P_1P_2P_3$ is a triangle and Q_1Q_2 is an edge. If Q_2 is in the shadow of $P_1P_2P_3$ blocking light from the source Q_1, the edge Q_1Q_2 intersect with the triangle $P_1P_2P_3$. In other words, if Q_1Q_2 intersect with $P_1P_2P_3$, following four conditions should be satisfied:

- the point Q_2 and P_3 are on the same side of the plane $Q_1P_1P_2$
- the point Q_2 and P_1 are on the same side of the plane $Q_1P_2P_3$
- the point Q_2 and P_2 are on the same side of the plane $Q_1P_3P_1$
- the point Q_1 and Q_2 are on different sides of the plane $P_1P_2P_3$.

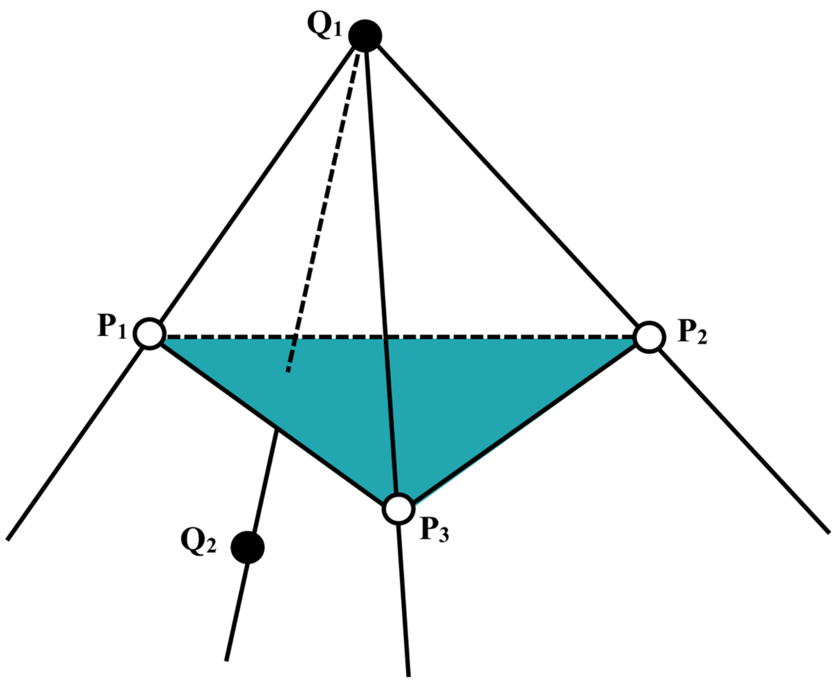


Fig 3. Diagram of an edge intersects with a triangle.

https://doi.org/10.1371/journal.pone.0184206.g003

Algorithm 2 shows the whole process to detect whether an edge intersects with a triangle.

```
Algorithm 2 The algorithm to decide whether an edge intersects with a triangle.
Input: three vertices of a triangle $P_1$, $P_2$, $P_3$ and two endpoints of an edge $Q_1$, $Q_2$
  $\vec{n_1} = \overrightarrow{Q_1P_1} \times \overrightarrow{Q_1P_2}$
  if dot_product ($\overrightarrow{Q_1P_3}, \vec{n_1}$) * dot_product ($\overrightarrow{Q_1Q_2}, \vec{n_1}$) < 0 then
    return False (False means no intersection and True means that the edge intersects with the triangle)
  else
    $\vec{n_1} = \overrightarrow{Q_1P_2} \times \overrightarrow{Q_1P_3}$
    if dot_product ($\overrightarrow{Q_1P_1}, \vec{n_1}$) * dot_product ($\overrightarrow{Q_1Q_2}, \vec{n_1}$) < 0 then
      return False
    else
      $\vec{n_1} = \overrightarrow{Q_1P_3} \times \overrightarrow{Q_1P_1}$
      if dot_product ($\overrightarrow{Q_1P_2}, \vec{n_1}$) * dot_product ($\overrightarrow{Q_1Q_2}, \vec{n_1}$) < 0 then
        return False
      else
        $\vec{n_1} = \overrightarrow{P_1P_1} \times \overrightarrow{P_1P_3}$
        if dot_product $\overrightarrow{Q_1P_1}, \vec{n_1}$) * dot_product ($\overrightarrow{Q_2P_1}, \vec{n_1}$) > 0 then
          return False
        else
          return True
        end if
      end if
    end if
  end if
```

For large molecules, the amount of points and triangles are huge. If we do smoothing for all points every round, the computation is expensive. To reduce the computation complexity, we use a screening method. Every round doing the modified Laplacian smoothing, we just choose the vertices and the neighbored vertices of the low-quality triangles (the triangles with low edge ratio or containing tiny angles). The detailed algorithm is shown in algorithm 3. Firstly, we do modified Laplacian smoothing for all the points once. Then we find out all the triangles with low-quality. The vertices and the neighbored vertices of these low-quality triangles are considered as the target points which need to be smoothed next round. And we do smoothing to these points several rounds, usually three or four rounds. After this, we will find out the new irregular triangles from the targets we got last round. From the irregular triangles, we can get the new target points which should be smoothed next round. Repeating in this manner, the points which need to be smoothed are getting fewer and fewer. In this way, we don't need to do modified Laplacian smoothing for all the points in the mesh every round, which reduces the computational cost greatly. When the quality of the mesh satisfies preset goal, the smoothing process is ended. Usually, we control the number of rounds of doing modified Laplacian smoothing for irregular triangles. In our program, the default rounds is set to 100. However, in most cases, the smoothing is convergent before reaching 100 rounds.

Algorithm 3 The algorithm of mesh smoothing using modified Laplacian method.

```
for each point $P$ in the initial mesh do
  do modified Laplacian smoothing to $P$
end for
let empty set $L$ be a temporary point set
for each triangle $T$ in the latest mesh do
  if the quality of $T$ is required to be improved then
    add the vertices of $T$ and all the neighbour vertices of $T$ to the point set $L$
  end if
end for
while TRUE do
  do modified Laplacian smoothing to the point in point set $L$ several rounds
repeatedly construct a new temporary point set $L'$
  find all the singular triangles in $L$ and add the vertices of the triangles
and the neighbored vertices to $L'$
  set $L = L'$
  if rounds of doing modified Laplacian smoothing exceeds presupposed number
  then
    exit
  end if
end while
```

Results

For large molecules, some surface meshes generated by TMSmesh 2.0 can not be directly used to boundary element method (BEM) simulation or generation of body-fitted volume mesh due to singular triangles with tiny angles or very short edges. The algorithms introduced in the Method section are used to improve the quality of these meshes. We test the initial and improved meshes in BEM computations of Poisson-Boltzmann (PB) electrostatics and generating corresponding surface conforming volume meshes that are required in FEM simulations. The BEM software used is a publicly available PB solver, AFMPB [37]. In FEM simulations, the corresponding volume meshes are generated by TetGen [38]. Table 1 shows some examples which demonstrate that our improvement algorithms are effective. The number of atoms of the test molecular ranges from several thousand to several hundred thousand. The PQR Benchmark can be downloaded from https://doi.org/10.6084/m9.figshare.5349229.v1. The

Table 1. Results of applying the initial meshes and the improved meshes to AFMPB and TetGen.

Molecule	Natoms	Number of vertices		Number of triangles		AFMPB (solvation energy)		TetGen	
		Initial	Improved	Initial	Improved	Initial	Improved	Initial	Improved
AChE monomer	8280	67955	63361	135702	126782	-NAN	-2.50917e+03	Failed	Success
connexin	19884	213362	210219	426716	420778	-NAN	1.80728e+03	Failed	Success
AChe tetramer	36638	381665	366227	762514	732634	-NAN	-7.30957e+03	Failed	Success
30S ribosome	88431	489315	480332	979094	961920	-NAN	-2.14092e+06	Failed	Success
70S ribosome	165337	1160599	1147035	2324466	2298102	-NAN	-3.42853e+06	Failed	Success
3K1Q	203135	975340	925914	1948856	1853892	-NAN	-7.0773e+04	Failed	Success

https://doi.org/10.1371/journal.pone.0184206.t001

initial molecular surface meshes are provided by TMSmesh 2.0. They are manifold meshes but contains highly irregular and singular triangles. The algorithms introduced in the Method section can remove the redundant points and improve the qualities of triangles effectively. The meshes are smoothed 100 rounds. The improved meshes can be applicable in AFMPB [37] and TetGen [38] directly. The volume mesh generated by TetGen can used in finite element method simulations, such as Ichannel [39], SMPBS [40], mFES [41] and so on.

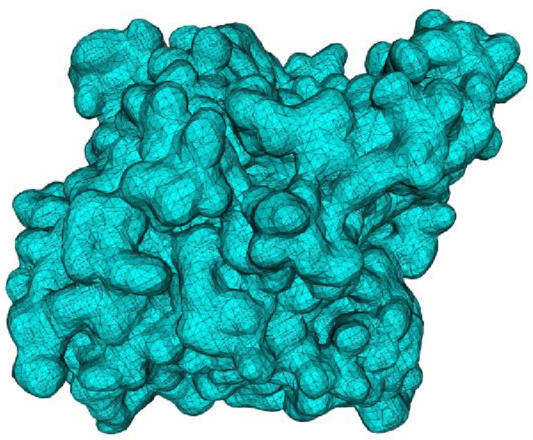
Table 2 shows the CPU time for mesh quality improvement. All computations run on a computer with Intel® Xeon® CPU E5-4650 v2 2.4GHz and 126GB memory under 64bit Linux system. Before deleting the redundant points and doing Laplacian smoothing, we first check whether the mesh is manifold and delete the small cavities inside the molecules. If we reserve the small cavities, they tend to collapse after several times of Laplacian smoothing. The fourth column in Table 2 shows the time cost by checking manifoldness and deleting cavities. The fifth column shows the CPU time cost by deleting the redundant points and doing Laplacian smoothing.

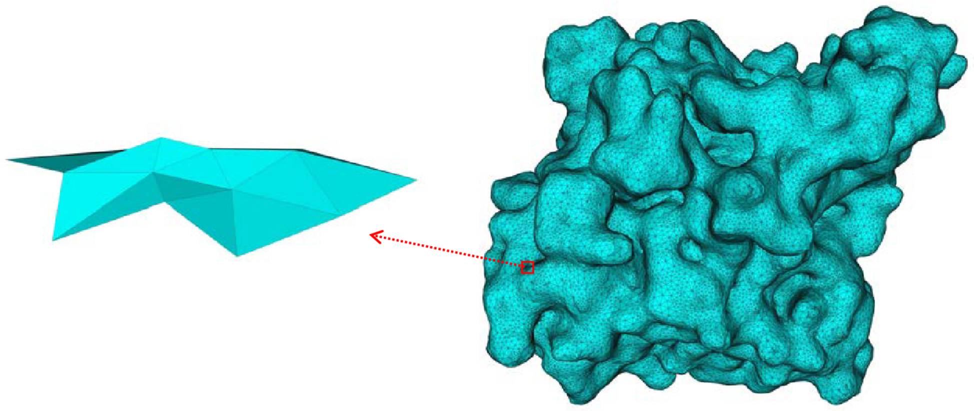
We compared the mesh quality of the improved meshes by our SMOPT with those by ISO2mesh [42]. ISO2mesh is a free matlab/octave-based mesh generation and processing toolbox. We use the meshresample and smoothsurf modules in ISO2mesh to improve the initial meshes. To make the improved meshes from ISO2mesh comparable to those from SMOPT, in the meshresample module, we set the vertex number in output meshes from ISO2mesh close to those from SMOPT. In the smoothsurf module of ISO2mesh, the Laplacian smoothing method is chosen and the meshes are smoothed 100 rounds. In addition, we also do the same test in MeshLab [43]. MeshLab is an open source system for processing and editing 3D triangular meshes. It provides a set of tools for editing, cleaning, healing, inspecting, rendering, texturing and converting meshes. We use MeshLab to do Laplacian smoothing for the meshes generated by TMSmesh 2.0. However, after several rounds of smoothing, some points with the coordinates (NaN, NaN, NaN) appear, which prevents us to do further test. So, here we only show the results from SMOPT and ISO2mesh. Fig 4 shows the initial and improved surface

Table 2. CPU time for mesh quality improvement by SMOPT.

Molecule	Number of vertices		CPU Time (s)	
	Initial	Improved	check manifold and delete small cavities inside	delete redundant points and do Laplacian smoothing
AChE monomer	67955	63361	3.39	43.06
connexin	213362	210219	21.04	201.52
AChe tetramer	381665	366227	59.22	364.55
30S ribosome	489315	480332	97.26	221.12
70S ribosome	1160599	1147035	447.80	580.70
3K1Q	975340	925914	359.49	465.84

https://doi.org/10.1371/journal.pone.0184206.t002





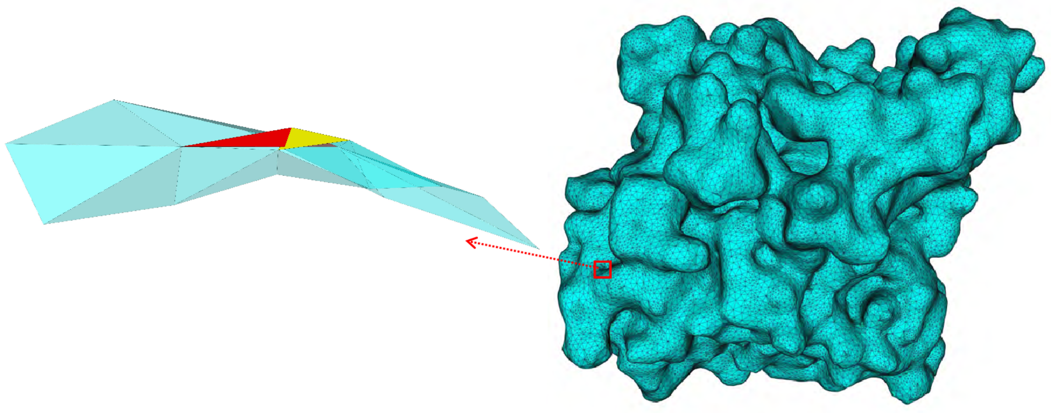


Fig 4. Initial mesh and improved meshes for an AChE monomer. (a) is the initial mesh, (b) is the improved mesh by SMOPT and (c) is the improved mesh by ISO2mesh. The left sub-figure in (c) shows an example of self-intersection in the mesh improved by ISO2mesh. The red triangle intersect with the yellow triangle. The left sub-figure in (b) shows the same region as the sub-figure in (c) and there is no intersection.

https://doi.org/10.1371/journal.pone.0184206.g004

Table 3. Number of non-manifold errors in meshes produced by SMOPT and ISO2mesh.

Molecule	Number of vertices			Number of intersecting triangle pairs		
	Initial	Improved		Initial	Improved	
		SMOPT	ISO2mesh		SMOPT	ISO2mesh
AChE monomer	67955	63361	63341	0	0	57
connexin	213362	210219	207126	0	0	10
AChE tetramer	381665	366227	366262	0	0	256
30S ribosome	489315	480332	480503	0	0	317
70S ribosome	1160599	1147035	1146652	0	0	1316
3K1Q	975340	925914	925640	0	0	351

https://doi.org/10.1371/journal.pone.0184206.t003

meshes by these two methods for an enzyme molecule AChE monomer. Compared to the initial mesh, both improved meshes are smoother and become closer to the molecular surface, and the two improved meshes are similar in the aspect of visualization. But there are self-intersections in the mesh improved by ISO2mesh. An example is shown in Fig 4. Table 3 compares mesh quality in aspect of manifoldness. There is no self-intersection in the meshes produced by SMOPT while the improved meshes by ISO2mesh can not preserve the manifoldness of the original meshes.

Fig 5 shows the areas and volumes of the initial meshes and the improved meshes generated by SMOPT and ISO2mesh. The areas of the improved meshes by SMOPT has small deviations from those of the initial meshes. Many roughed regions are smoothed after improvement, which will reduce the surface area. The volume of the smoothed mesh is almost preserved and keeps a deviation within 3%. ISO2mesh preserves both the mesh area and volume. An important reason is that ISO2mesh re-samples the vertices while SMOPT doesn't. We will consider to add a re-sample module into SMOPT in the future.

We also compared the uniformness of the meshes. The distribution of the ratios of the shortest edge length to the longest edge length of each triangle and the distribution of the angles of each triangle are used to describe the uniformness of a triangulated surface mesh. A

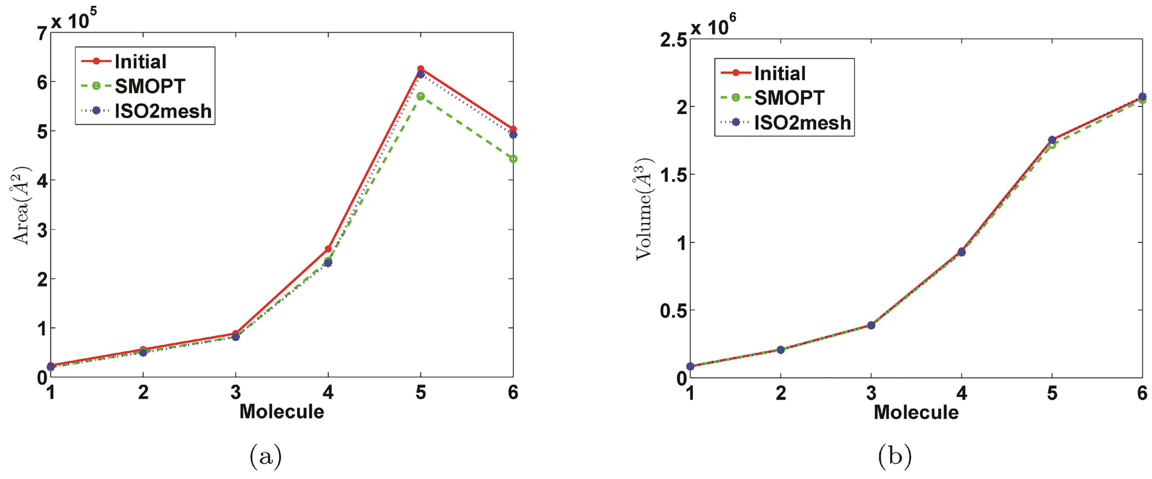


Fig 5. Comparison of area and volume between the initial meshes and the improved meshes generated by SMOPT and ISO2mesh for molecules in Table 3. (a) Area (b) Volume.

https://doi.org/10.1371/journal.pone.0184206.g005

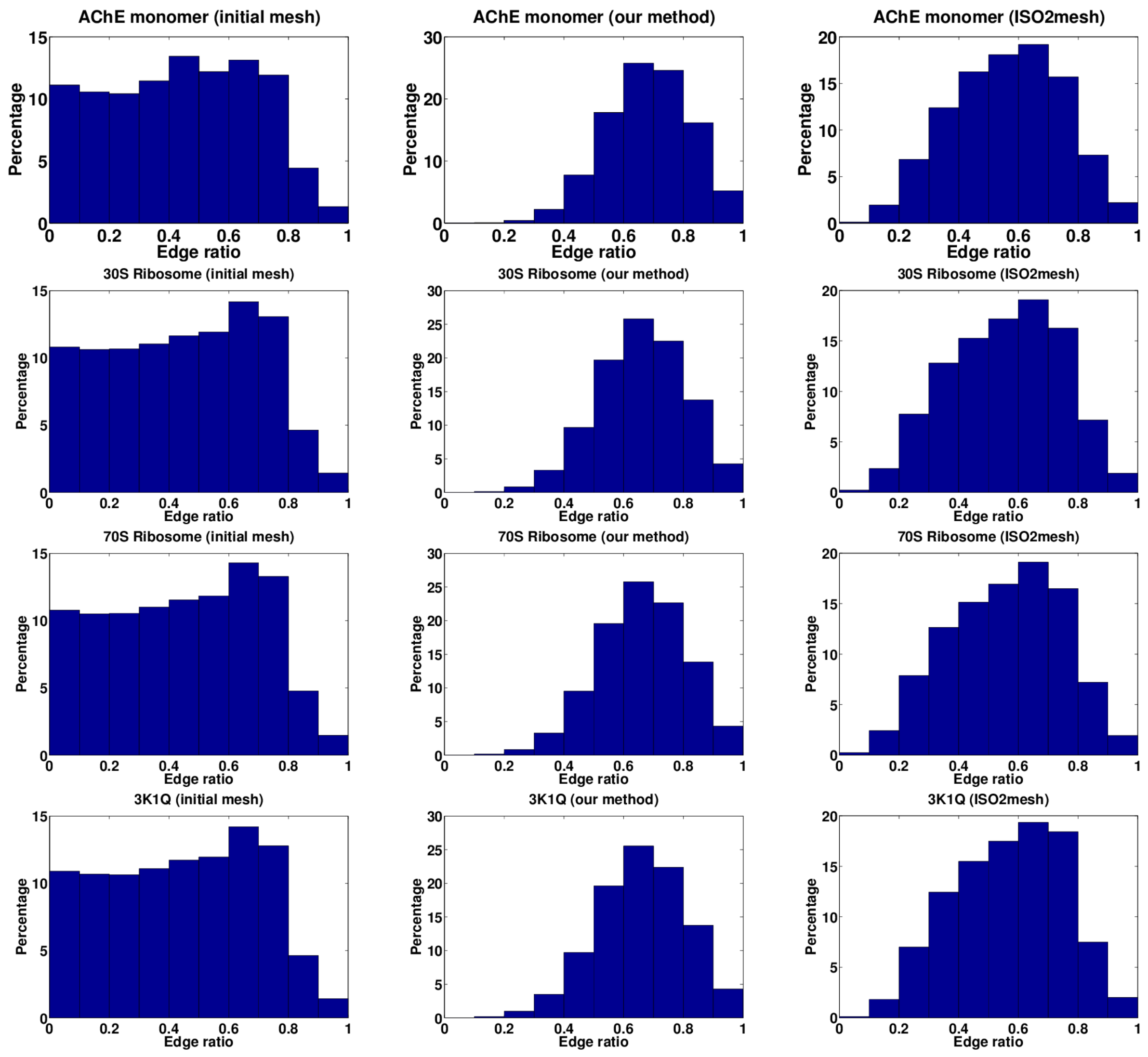


Fig 6. Distributions of ratio of the shortest edge length to the longest edge. The left column is the ratios of the initial mesh, the middle column is the ratios of the improved mesh by our method and the right column is the ratios of the improved mesh by ISO2mesh.

https://doi.org/10.1371/journal.pone.0184206.g006

ratio of 1.0 corresponds to an equilateral triangle, and a ratio close to 0 is indicative of a very poor uniformness. That is, the higher the ratio, the better the quality of the triangle. Fig 6 shows that the edge ratios of the initial meshes are uniform distributed between (0, 0.8). The edge ratios of the improved meshes by our method SMOPT and ISO2mesh are all clustered around 0.7, but the results by SMOPT are more concentrated than ISO2mesh. Additionally, there is no triangle whose edge ratio is between 0 and 0.1 in the improved meshes by our method, which means that there are few triangles with very poor quality after improvement. The angle distributions of surface triangles in the initial and improved meshes are shown in Fig 7. The angle distribution of the improved meshes are clustered around 40° to 80° and a small portion are close to 0° or 180°, which also indicates that the improved meshes have few sharp angles.

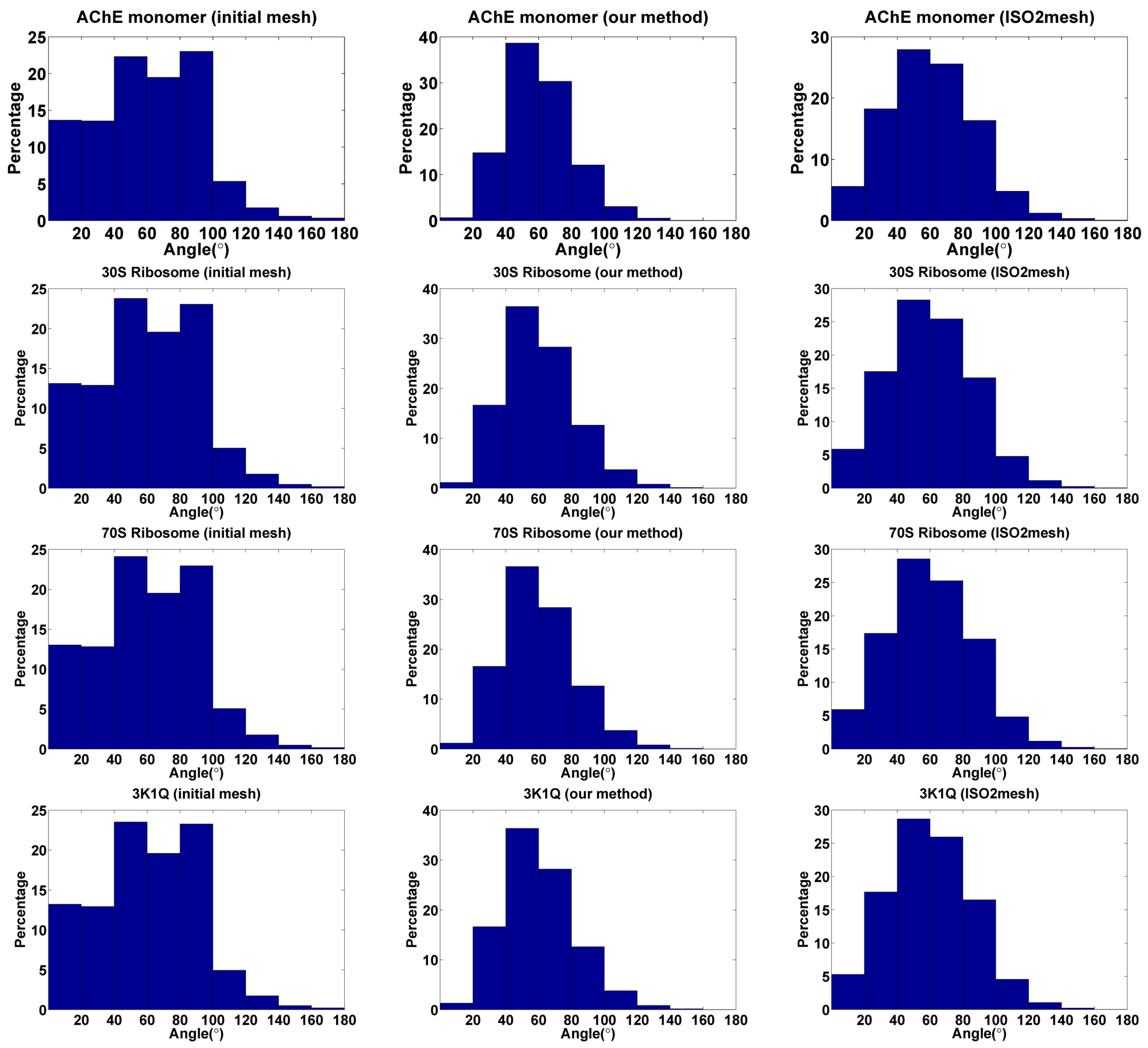


Fig 7. Distributions of angles of each triangle. The left column is the distributions of the initial mesh, the middle column is the distributions of the improved mesh by our method and the right column is the distributions of the improved mesh by ISO2mesh.

https://doi.org/10.1371/journal.pone.0184206.g007

Conclusion

In this paper, we have described a systematic procedure, SMOPT, to remove redundant points in a 3D triangular surface mesh and to smooth the mesh using a modified Laplacian smoothing method. The improved meshes are shown to be of good quality and can be applied to numerical simulation successfully. The program SMOPT can be used together with or integrated into a surface mesh generator, such as TMSmesh 2.0. SMOPT can be also used as a standalone software and is downloadable at https://doi.org/10.6084/m9.figshare.5346169.v1 and can be run online at http://www.xyzgate.com.

Acknowledgments

We thank the referees for their helpful comments.

Author Contributions

Conceptualization: Tiantian Liu, Minxin Chen, Yu Song, Hongliang Li, Benzhuo Lu.

Data curation: Tiantian Liu, Minxin Chen, Benzhuo Lu.

Formal analysis: Tiantian Liu, Minxin Chen, Benzhuo Lu.

Funding acquisition: Minxin Chen, Yu Song, Hongliang Li, Benzhuo Lu.

Investigation: Tiantian Liu, Minxin Chen, Benzhuo Lu.

Methodology: Tiantian Liu, Minxin Chen, Benzhuo Lu.

Project administration: Minxin Chen, Yu Song, Hongliang Li, Benzhuo Lu.

Resources: Tiantian Liu, Minxin Chen, Benzhuo Lu.

Software: Tiantian Liu, Minxin Chen.

Supervision: Minxin Chen, Benzhuo Lu.

Validation: Tiantian Liu, Minxin Chen, Yu Song, Hongliang Li, Benzhuo Lu.

Visualization: Tiantian Liu, Minxin Chen.

Writing – original draft: Tiantian Liu, Minxin Chen, Benzhuo Lu.

Writing – review & editing: Tiantian Liu, Minxin Chen, Benzhuo Lu.

References

1. Lu BZ, Zhou YC, Holst MJ, MaCammon JA. Recent progress in numerical methods for the Poisson-Boltzmann equation in biophysical applications. Commun in Comput Phys. 2008; 3(5):973–1009.
2. Marchandise E, Remacle JF, Geuzaine C. Optimal parametrizations for surface remeshing. Engineering with Computers. 2014; 30(3):383–402. https://doi.org/10.1007/s00366-012-0309-3
3. Alliez P, Ucelli G, Gotsman C, Attene M. Recent Advances in Remeshing of Surfaces. In: Shape Analysis and Structuring. Springer; 2008. p. 53–82.
4. Borouchaki H, Laug P, George PL. Parametric surface meshing using a combined advancing-front generalized Delaunay approach. International Journal for Numerical Methods in Engineering. 2000; 49:233–259.
5. Laug P, Borouchaki H. Interpolating and meshing 3D surface grids. International Journal for Numerical Methods in Engineering. 2003; 58(2):209–225. https://doi.org/10.1002/nme.770
6. Attene M, Falcidieno B, Spagnuolo M, Wyvill G. A mapping-independent primitive for the triangulation of parametric surfaces. Graphical Models. 2003; 65(5):260–273. https://doi.org/10.1016/S1524-0703(03)00048-1
7. Marcum DL. Efficient Generation of High-Quality Unstructured Surface and Volume Grids. Engineering With Computers. 2001; 17(3):211–233. https://doi.org/10.1007/PL00013386
8. Tristano JR, Owen SJ, Canann SA. Advancing front surface mesh generation in parametric space using Riemannian surface definition. In: Proceedings of 7th international meshing roundtable; 1998. p. 429–455.
9. Zheng Y, Weatherill NP, Hassan O. Topology abstraction of surface models for three-dimensional grid generation. Engineering With Computers. 2001; 17(1):28–38. https://doi.org/10.1007/s003660170021
10. Surazhsky V, Alliez P, Gotsman C. Isotropic Remeshing of Surfaces: A Local Parameterization Approach. In: Proceedings of 12th international meshing roundtable; 2003. p. 215–224.
11. Béchet E, Cuillière Jean Christophe T F. Generation of a finite element mesh from stereolithography (STL) files. Computer-Aided Design. 2002; 34(1):1–17. https://doi.org/10.1016/S0010-4485(00)00146-9
12. Ito Y, Nakahashi K. Direct Surface Triangulation Using Stereolithography Data. AIAA Journal. 2002; 40(3):490–496. https://doi.org/10.2514/3.15087
13. Wang D, Hassan O, Morgan K, Weatherill N. Enhanced remeshing from stl files with applications to surface grid generation. Commun Numer Meth Eng. 2007; 23:227–239. https://doi.org/10.1002/cnm.894

14. Lévy B, Liu Y. Lp Centroidal Voronoi Tesselation and its Applications. ACM Transactions on Graphics. 2010; 29(4):119:1–119:11.

15. Dassi F, Si H. In: A Curvature-Adapted Anisotropic Surface Re-meshing Method. Springer International Publishing;. p. 19–41.

16. Alliez P, Meyer M, Desbrun M. Interactive Geometry Remeshing. ACM Trans Graph. 2002; 21(3): 347–354. https://doi.org/10.1145/566654.566588

17. Eck M, DeRose T, Duchamp T, Hoppey H, Lounsberyz M, Stuetzle W. Multiresolution Analysis of Arbitrary Meshes. In: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques; 1995. p. 173–182.

18. Floater MS. Parametrization and smooth approximation of surface triangulations. Computer Aided Geometric Design. 1997; 14(3):231–250. https://doi.org/10.1016/S0167-8396(96)00031-3

19. Floater MS. Mean value coordinates. Computer Aided Geometric Design. 2003; 20(1):19–37. https://doi.org/10.1016/S0167-8396(03)00002-5

20. Lévy B, Petitjean S, Ray N, Maillot J. Least Squares Conformal Maps for Automatic Texture Atlas Generation. In: ACM SIGGRAPH conference proceedings; 2002. p. 362–371.

21. Tutte W. How to draw a graph. In: Proceedings of the London Mathematical Society. vol. 13; 1963. p. 743–768. https://doi.org/10.1112/plms/s3-13.1.743

22. Hormann K, Greiner G. Mips: an efficient global Parametrization method. In: Curve and surface design; 2000.

23. Sheffer A, Lévy B, Lorraine I. Abf++: fast and robust angle based flattening. ACM Transactions on Graphics. 2005; 24(2):311–330. https://doi.org/10.1145/1061347.1061354

24. Zigelman G, Kimmel R. Texture mapping using surface flattening via multi-dimensional scaling. IEEE Trans Vis Comput Graph. 2002; 8:198–207. https://doi.org/10.1109/2945.998671

25. Ben-Chen M, Gotsman C, Bunin G. Conformal flattening by curvature prescription and metric scaling. Comput Graphics Forum. 2008; 27(2):449–458. https://doi.org/10.1111/j.1467-8659.2008.01142.x

26. Zayer R, Lévy B, Seidel HP. Linear angle based parameterization. In: Proceedings of ACM/EG symposium on geometry processing conference; 2007.

27. Frey PJ. About surface remeshing. In: Proceedings of 9th International Meshing Roundtable; 2000. p. 123–136.

28. Frey PJ, Borouchaki H. Geometric surface mesh optimization. Computing and Visualization in Science. 1998; 1:13–121. https://doi.org/10.1007/s007910050011

29. Hoppe H. Progressive meshes. Computer Graphics. 1996; 30:99–108.

30. Rassineux A, Villon P, Stab O. Surface remeshing by local hermite diffuse interpolation. International Journal for Numerical Methods in Engineering. 2000; 49(1-2):31–49. https://doi.org/10.1002/1097-0207(20000910/20)49:1/2%3C31::AID-NME921%3E3.0.CO;2-6

31. Chen MX, Lu BZ. TMSmesh: A Robust Method for Molecular Surface Mesh Generation Using a Trace Technique. J Chem Theory Comput. 2011; 7(1):203–212. https://doi.org/10.1021/ct100376g PMID: 26606233

32. Chen MX, Tu B, Lu BZ. Triangulated Manifold Meshing Method Preserving Molecular Surface Topology. Journal of Molecular Graphics and Modelling. 2012; 38(1):411–418. https://doi.org/10.1016/j.jmgm.2012.09.006 PMID: 23117290

33. Liu T, Chen M, Lu B. Efficient and Qualified Mesh Generation for Gaussian Molecular Surface Using Adaptive Partition and Piecewise Polynomial Approximation. https://arxivorg/pdf/161103474pdf. 2017;.

34. Liu T, Chen M, Lu B. Parameterization for molecular Gaussian surface and a comparison study of surface mesh generation. Journal of Molecular Modeling. 2015; 21(5). https://doi.org/10.1007/s00894-015-2654-9

35. Herrmann LR. Laplacian-isoparametric grid generation scheme. Journal of the Engineering Mechanics Division. 1976; 102(5):749–756.

36. Sorkine O, Lipman Y, Cohen-Or D, Spencer S. Laplacian Surface Editing. In: Proceedings of the 2004 Eurographics; 2004.

37. Zhang B, Peng B, Huang J, Pitsianis NP, Sun X, Lu- B. Parallel AFMPB solver with automatic surface meshing for calculations of molecular solvation free energy. Computer Physics Communications. 2015; 190:173–181. https://doi.org/10.1016/j.cpc.2014.12.022

38. Si H. TetGen, a Delaunay-based quality tetrahedral mesh generator. ACM Transactions on Mathematical Software. 2015; 41(2):Article 11. https://doi.org/10.1145/2629697

39. Tu B, Bai S, Chen M, Xie Y, Zhang L, Lu B. A software platform for continuum modeling of ion channels based on unstructured mesh. Computational Science & Discovery. 2014; 7(1):014002. https://doi.org/10.1088/1749-4680/7/1/014002

40. Xie Y, Ying J, Xie D. SMPBS: Web server for computing biomolecular electrostatics using finite element solvers of size modified Poisson-Boltzmann equation. Journal of Computational Chemistry. 2017; 38(8):541–552. https://doi.org/10.1002/jcc.24703 PMID: 28052351

41. Sakalli I, Schöberl J, Knapp EW. mFES: A Robust Molecular Finite Element Solver for Electrostatic Energy Computations. J Chem Theory Comput. 2014; 10(11):5095–5112. https://doi.org/10.1021/ct5005092 PMID: 26584389

42. Fang Q, Boas D. Tetrahedral mesh generation from volumetric binary and gray-scale images. In: Proceedings of IEEE International Symposium on Biomedical Imaging 2009; 2009. p. 1142–1145.

43. Cignoni P, Callieri M, Corsini M, Dellepiane M, Ganovelli F, Ranzuglia G. MeshLab: an Open-Source Mesh Processing Tool. In: Sixth Eurographics Italian Chapter Conference; 2008. p. 129–136.