ALGORITHMS FOR MESHING SMOOTH SURFACES AND THEIR VOLUMES

BY

SHI XINWEI

B.S., Harbin Institute of Technology, 1998

M.S., Harbin Institute of Technology, 2000

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

2006

To my wife Ei Ei.

Acknowledgements

I owe special thanks to many people throughout the preparation of this thesis for their guidance, support, help and encouragement. First and foremost, I would like to thank my supervisor Dr. Cheng Ho-lun for his invaluable help and guidance presented in various aspects of the work. I got introduced to the field of computational geometry when I met him for an introductory discussion of my Ph.D study. The following numerous discussion sessions with him enlightened me in pursuing further research in the field. His constructive feedback while working on manuscripts improved my writing skills. His encouragement always pushed my progressing forward when I felt frustrated. I also thank him for the care of my life, and the generous dinners and coffee.

I am grateful to my other committee members Associate Professor Tan Tiow Seng and Associate Professor Chionh Eng Wee for their help and guidance at different stages of my thesis. Special thanks to Tiow Seng for his effort to maintain a good research environment in computer graphics research lab. His meticulous attention to detail and rigorous scholarship also motivate me to work harder. I also thank Dr. Huang Zhiyong for giving me valuable advice on academic matters and career options.

I am extremely fortunate to have the inspiring discussions with Professor Herbert Edelsbrunner during the SoCG conference. I am also grateful to Professor Tien-Tsin Wong for the valuable advice and discussions. Thanks to Professor Siu-Wing Cheng for the constructive discussions when he visited us. I would like to thank Professor

Bernd Hamann, Professor Patrice Koehl, Professor Nina Amenta, Professor Kwan-Liu Ma, and Dr. Vijay Natarajan for the wonderful discussions when I visited UC Davis.

During my stay in NUS, I have enjoyed the friendship with many people. I want to thank all of them for the fancy time we spent together. Special thanks go to Chen Chao and Tony Tan for the joyful talks in the afternoon break. I also thank my labmates Zhao Yonghong, Xiao Yongguan, Liang Yongqi, Rong Guodong, Calvin Lim, Ng Chu Ming, Zhang Xia, Yu Hang, and Zhang Xin for making the office a nice place to stay.

Last but not the least, I would like to convey my thanks to my parents Shi Tianshun and Huang Xiuzheng for their love and support throughout my life. I am deeply grateful to my wife Ei Ei for her endless love, perpetual support, cheering encouragement and strong confidence in me. She deserves the particular recognition for being the driving force in my life. Finally, I would like to thank our forthcoming baby who will be arriving in this world soon. This special excitement has greatly inspired me in accomplishing this thesis.

Table of Contents

Abstract

Quality meshes of molecular models are essential to support computational tools for new drug discovery. However, it is still challenging to generate the meshes efficiently. The principal goal of this thesis was to develop and implement efficient algorithms for triangulating the molecular skin surface and their bounded volumes with guaranteed quality.

Two skin surface meshing algorithms were developed, namely, the adaptive sweeping skin meshing algorithm and the Delaunay skin meshing algorithm. The first algorithm adapts the advancing front method to sweep the surface mesh from the bottom to the top of the skin surface until the whole surface is covered. In particular, the algorithm employs Morse theory to handle the front collision problem in advancing front meshing. As such, the algorithm improves the efficiency of skin meshing dramatically. Moreover, the mesh quality and the homeomorphism between the triangulation and the surface are guaranteed as well. The second meshing algorithm incrementally samples points on the surface and constructs the Delaunay triangulation simultaneously. By associating each sample point to a ball centered on the surface, the algorithm achieves an even ε-sampling of the skin surface when it terminates. The restricted Delaunay triangulation, a subset of the Delaunay triangulation of the ε-sampling, forms a quality mesh of the skin surface. This second algorithm not only offers guarantees on both the mesh quality and the homeomorphism between the triangulation and the skin surface but also performs excellently in practice.

Based on the result of quality skin surface meshing, an algorithm for generating quality tetrahedral meshes of the volumes bounded by skin surfaces was developed. The algorithm applies the Delaunay refinement to a tetrahedral mesh bounded by the surface. In particular, the circumcenters of bad shape tetrahedra are inserted iteratively with a priority parameterized by its distance from the surface. The algorithm achieves an upper bound on radius-edge ratio of the tetrahedral mesh after the refinement. Moreover, the slivers are removed by assigning weight to the mesh vertices in a post processing procedure.

The implementation results provide evidence of the efficiency and quality guarantees of the algorithms. The skin meshes generated by the algorithms will serve as an essential component in the study of the molecular shape and functions.

List of Tables

List of Figures

Chapter 1

Introduction

Discovering new medicines or drugs for the treatment of diseases and improvement of people's living quality is one of the most important scientific challenges. Two keys in a successful drug discovery are the identification of the right cellular target, usually a protein molecule and the selection of the right drug candidates. A potent drug is a small molecule called a ligand that simultaneously optimizes its affinity with the target, and decreases the interaction between the ligand and other targets that could lead to side effects [73]. This process of target identification and drug selection usually involves large-scale experimental investigations, which results in lengthy and expensive drug discovery . For example, bringing up new medicine from the laboratory to the pharmacy takes an average of ten to fifteen years [65].

Computational tools that predict the interactions between proteins and ligands, namely, protein-ligand docking programs, accelerate the drug development process significantly. The docking problem has attracted great attention from computer scientists as well as biochemists because the protein-ligand interactions are largely characterized by the complementarity of their geometric shapes and chemical properties [67, 94]. A geometric formulation of the docking problem is as following. Given two proteins A and B, compute the alignment such that their shapes best complement each other. Three main issues are involved here, (i) developing suitable shape representations of the proteins to capture the shape features; (ii) searching the conformation space for the alignments of two proteins with complementary shape matching; (iii) evaluating the generated alignments to reduce false predictions. These three components are mutually correlated.

In particular, the molecular shape representation is the base of the algorithms for alignments searching and evaluation. On one hand, accurate shape representations are likely to improve fidelity of the generated alignments in procedure (ii). On the other hand, polygonal meshes for the molecular shapes facilitate accurate approximations of the chemical properties such as electrostatics potential to reduce the false alignments in procedure (iii) correctly [53, 67]. Although a number of docking programs have been studied, their prediction accuracy is still not sufficient for the application in the drug discovery process [94]. The low accuracy is partly due to the unsatisfactory molecular shape representations and the challenges in converting the continuous shape representation to discrete form.

This thesis develops efficient algorithms for building digital models of macromolecules such as proteins and DNAs using a new shape representation, namely, the *skin surface* defined by Edelsbrunner [37, 38]. I will focus on meshing the skin surfaces and their volumes with guaranteed quality. The surface mesh provides an accurate and efficient molecular shape representation that facilitates fast alignments searching algorithm. The volumetric mesh facilitates the approximation of the electrostatic potentials using the finite element methods. Applying the meshes in the protein ligand docking study will improve the prediction accuracy. Moreover, skin surfaces and meshing techniques are useful in various application areas other than protein-ligand docking, including geometric modeling, computer graphics, and mesh generation [40, 55, 61, 95]. Most of the techniques developed in this thesis are applicable to these other fields as well.

In the remainder of this chapter, I will first introduce the existing geometric models of proteins and justify the advantages of the skin model. Second, I will describe the needs of skin meshes for the protein-docking study and review the main meshing techniques. Finally, I will summarize the main contributions of this thesis.

1.1 Geometric Models of Proteins

Proteins are large molecules that typically consist of 500 to a few thousands atoms. Various interactions among these atoms such as chemical bonds and electrostatic forces result in a stable three dimensional structure of the protein, which defines a protein shape. Such three dimensional structure can be presented in various geometric models. Three different well-known protein models are illustrated in Figure 1.1. The

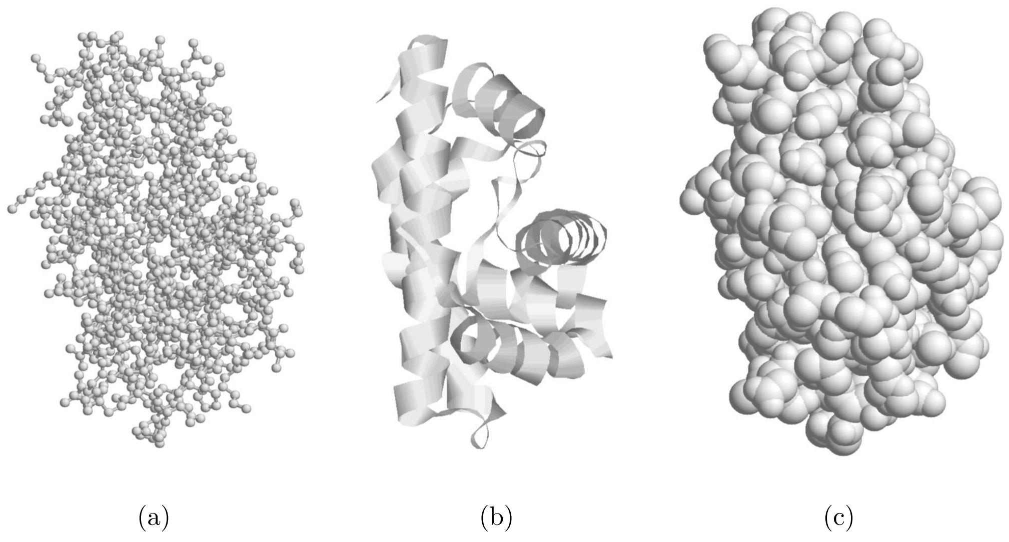


Figure 1.1: Three different geometric models for the protein Myoglobin. (a) Ball and stick model, (b) Cartoon model, (c) Space-filling model.

ball-stick model (Figure 1.1 (a)) represents a protein using a set of balls at the atomic centers connected by sticks corresponding to covalent bonds between pairs of atoms. Such representation emphasizes the chemical nature of the proteins. The Cartoon model (Figure 1.1 (b)) gives a high level view of the protein structure organization, in which the protein is considered as a folded chain of amino acids. It provides a simplified representation of a protein and is popular in applications such as protein folding [52]. The space-filling model, as shown in Figure 1.1 (c) represents a protein as a union of balls, in which each atom is modeled by a ball in $\mathbb{R}^3$ with its van der Waals radius. This representation shows the tight packing of the atoms in a protein.

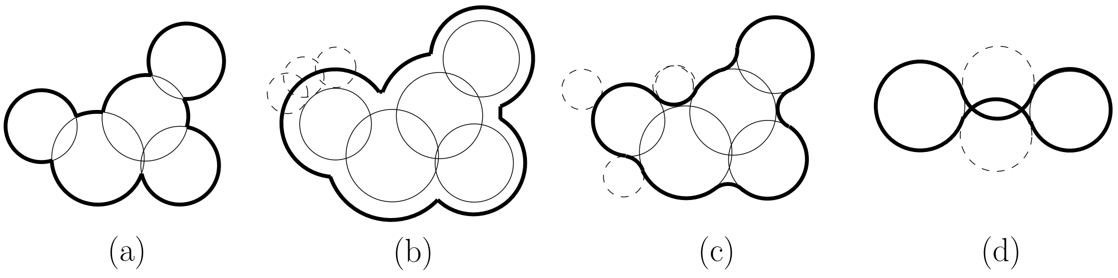


Figure 1.2: Existing molecular surface models. Dashed circles represent the probe sphere. (a) van der Waals surface (VW), (b) solvent accessible surface (SA), (c) molecular surface (MS), (d) self-intersection on molecular surfaces.

However, for computational purposes, especially for the study of the protein-ligand docking, surface models of proteins are more favorable since the key to the function of a protein is the existence of shape features such as depressions and protrusions on the boundary of the protein shapes, which are not characterized in the geometric models illustrated in Figure 1.1. There are three existing molecular surface models, that is, the van der Waals surface (or VW) model, the solvent accessible (or SA) model and the molecular surface (or MS) model [31]. See Figure 1.2. The VW model is defined as the boundary of the space-filling model of the molecule. The other two surface models are defined through tracing a probe sphere that rolls over the VW model. The SA model is the surface traced by the center of the probe sphere, while the MS model is the surface traced by the inward-facing surface of the probe sphere. The major advantage of the MS model over the other two is its smoothness in most cases. Because smooth surfaces can be meshed with good quality triangles, MS model facilitates accurate numerical computations [63]. However, this may not be possible because

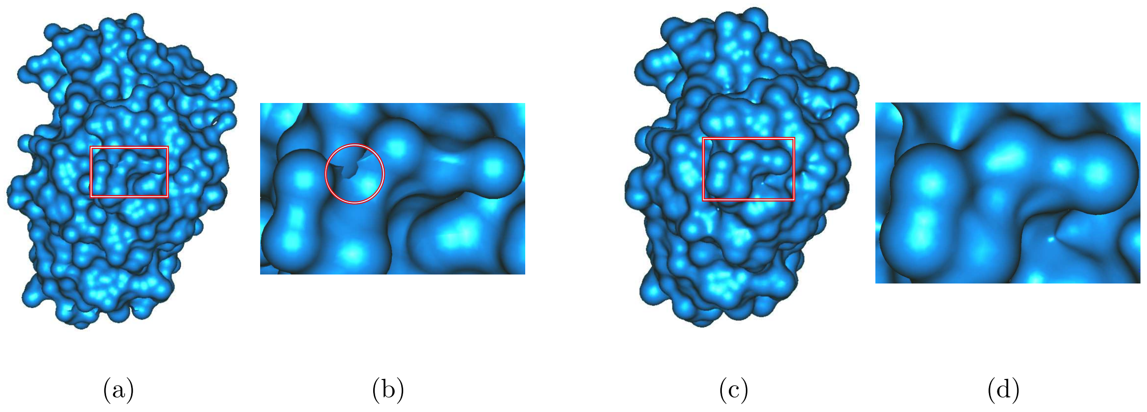


Figure 1.3: Comparison between the molecular surface model and the skin model for the protein Myoglobin.

sharp corners still exist since the MS model may have self-intersections, as illustrated in Figure 1.2 (d). On one hand, the cusp results in unfaithful representations of the molecules and unrobust meshing software implementations [5, 9, 96]. On the other hand, the self-intersections lead to singularities when computing the derivatives of the volume and area of molecular surface with respect to its atomic coordinates [45].

To circumvent these difficulties, we use a new shape representation, namely, the skin surface to model molecules. A skin surface is specified by a finite set of spheres and lends itself as a better surface model for molecules than the existing surface models. The self-intersection problem does not exist in molecular skin models as the skin surface is a C^1-continuous surface. See Figure 1.3 for a comparison of the molecular surface model and the skin model for a protein molecule. Figure 1.3 (a) shows the MS model of the protein Myoglobin and the cusps duo to self-intersections are highlighted in the rectangle. Figure 1.3 (b) illustrates the magnified view of the part with cusps. The corresponding skin model is illustrated in Figure 1.3 (c) and (d), which is smooth and free of any cusps. In addition, the skin surface also has a number of desirable properties for molecular modeling applications such as decomposability, complementarity and capability of free deformation, which will be introduced in Section 2.2.

1.2 Needs of Quality Skin Meshes

The skin model of proteins outperforms the existing surface models in terms of smoothness and other elegant properties [38]. Applying the skin model to protein-ligand docking investigations should improve the prediction accuracy of the docking programs. However, the skin surface is a continuous surface encoded in the functions specified by a set of spheres. In order to perform computations over the skin model, discrete forms of the surface are essential. Meshes are the most preferred discrete representations because they facilitate fast rendering for molecular visualization, geometric algorithms for shape feature extraction, and numerical methods for chemical properties computation. In the context of chemical properties computation, mesh quality are usually critical to the accuracy and convergency of the solution.

Surface Meshes. Skin surface meshes support molecular visualization applications. Since surface meshes can be rendered very efficiently by modern graphics hardware, the skin models can be visualized on the computer screen or virtual reality devices, which provide direct understanding and interaction of the molecular shapes. Figure 1.4 (a) shows an example of the rendered skin model of protein 1CHO. The surface meshes also support the visualization of the molecular properties, such as atomic charge, electrostatic potential, and polarization etc. The information can be encoded as color codes and texture maps over a mesh to represent these added dimensional properties.

In addition, skin surface meshes also facilitate efficient combinatorial algorithms to extract the shape features such as depressions and protrusions on the surface. These concave and convex features over the surface can be identified by computing the critical points of some real-valued functions defined on the surface. For example,

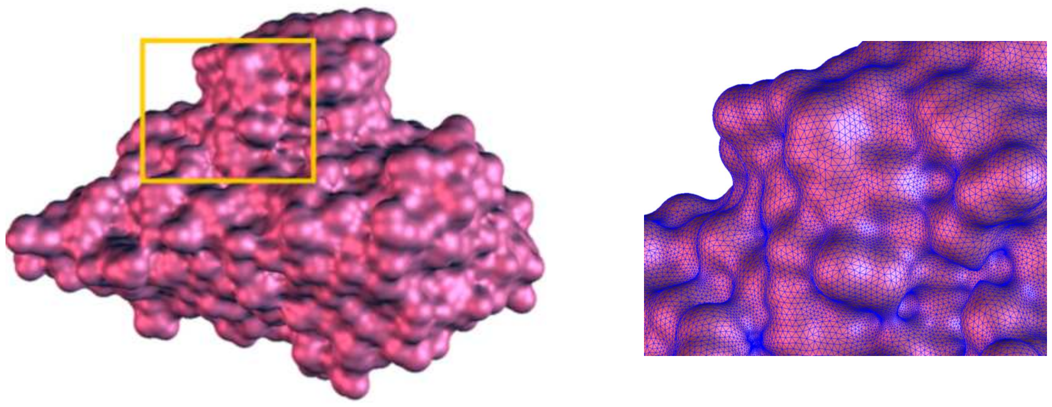

(a) (b)

Figure 1.4: The molecular skin model of the protein 1CHO and the zoomed in view of the partial mesh.

the critical points of the Connolly function used in [21] and the elevation function proposed by Agarwal et al. [4]. Since the critical points theory is originally developed on smooth surfaces and their critical points are hard to be efficiently computed, surface meshes can facilitate fast combinatorial algorithms for computing critical points on the base of an extension of the smooth concepts to the discrete analogs [44]. Thus, the shape complementarity computation in the protein docking can be materialized by matching the depressions and protrusions pairwisely. In addition, the accuracy of the mesh approximation affects the precision of the extracted features. Experimental results in [4] shows that an adaptive mesh approximations with guaranteed quality significantly reduces the number of noisy critical points of elevation functions. In which, the adaptiveness of the surface mesh means that the edge lengths in the mesh are adaptive to the local surface curvature and the guaranteed quality is defined by an lower bound of the minimum angle of the triangles in the mesh. See Figure 1.4 (b) for an example.

Volumetric Meshes. Volumetric meshes of the skin model are essential to improve the accuracy of the docking program. The searching algorithm based on shape complementarity usually generates a number of alignments that are potential solutions. Among these alignments, only one of them is the real docking conformation and the remainders are false positive alignments. To discriminate the real docking conformation from the set of potential solutions, a scoring function defined in terms of the biological and chemical properties is essential to rank the potential solutions. On the base of the observation that the interacting protein and ligand always exhibits excellent complementarity in the electrostatic potential, incorporation of the electrostatic potential in the scoring function would make it more reliable to filter out the false conformations. The Poisson-Boltzmann equation(PBE) is one of the most popular approach to model the electrostatic of large molecules [10]. Using the solution of PBE to predict the electrostatic property of molecules achieves good agreement with experimental results [63, 80]. Since the PBE is a non-linear partial differential equation, there is no analytical solutions for the PBE currently and it is necessary to use numerical methods, for example, finite element methods. The accuracy and stability of the solution with finite element methods depend on the quality of the elements used to decompose the molecular volume. Moreover, the solution of PBE is sensitive to the boundary of the molecular model [10]. As a result, a quality volumetric mesh of the molecule that conforms to its boundary is necessary for computing the molecular electrostatic by solving the PBE.

To conclude, quality meshes for the skin surfaces and the bounded volumes are essential for scientific computing in the study of protein ligand docking. However, the skin meshing problem is still far from being solved. Although Cheng et al. [22, 23] and Kruithof et al. [70, 71] had addressed the problem recently, both their work have deficiencies. Cheng's algorithms [23] generated topologically correct surface meshes with guaranteed quality but the efficiency is unsatisfactory. For instance, it takes hours to generate a skin surface of a protein molecule with about one thousand atoms. The algorithms presented by Kruithof et al. [70, 71] offered little in the way of guaranteeing the mesh quality. Moreover, the tetrahedralization problem of the skin volume is still open. Therefore, the skin meshing problem deserves further investigations.

Next, I will review the previous mesh generation techniques to identify the challenges and gain some new insights for skin meshing.

1.3 Meshing Techniques: A Brief Review

In scientific computing and engineering, decomposing a physical domain into a mesh of primitive elements is an essential step in a wide range of applications such as computer graphics and numerical simulations. This is referred to the problem of *mesh generation*. Mesh generation algorithms should guarantee that the output mesh elements have high shape quality so that the numerical simulations converge and achieve accurate solutions.

The most popular shapes of the mesh elements are triangles and tetrahedra in two and three dimensions respectively because they have several advantages such as the flexibility to fit complicated domains and ease of refinement over other types of meshes, for instance, the hexahedral meshes. Thus, I will focus on the meshing techniques for generating triangular and tetrahedral meshes. A number of substantial advances have been achieved in both theories and practices. Most of the previous work has been focused on meshing polygons in two dimensions [11, 30, 87], and polyhedra in three dimensions [28, 42, 78, 90]. A few works also has been proposed for meshing parametric surfaces [19, 33, 58, 97] and implicit surfaces [62, 6, 15, 60, 66].

Most of the mesh generation algorithms can be categorized into one of the three main approaches: *advancing front, Delaunay and quadtree/octree mesh generators.* There are certainly differences in the complexity and performance when applying these approaches to mesh polygons, polyhedra and smooth surfaces. I will sketch the essential ideas of each approach and justify its effectiveness and challenges for meshing the skin surface according to the pros and cons. For detailed review of the mesh generation algorithms, readers can refer to the recent survey paper by Bert and Plassmann [13], Owen [81] and Edelsbrunner [39].

Advancing-front Methods. Advancing front methods [25, 60, 66, 70, 89, 64] construct meshes from the domain boundary to the interior in a way of a layer by a layer. The boundaries of the domain are firstly discretized to a collection of edges (in two dimensions) or triangle faces (in three dimensions), which is called the *front*. Starting from one element in the front, new triangles or tetrahedra are added incrementally. At the same time, the front is updated and advancing towards the unmeshed region. The mesh is completed when the front becomes empty.

Advantages of advancing front methods include high efficiency, good mesh quality and ease of implementation. On the other hand, it is challenging to avoid the collision of the front elements during the advancing. Front collision leads to overlapping triangles in the mesh, which may fail the meshing procedure. An efficient way to handle the front collision problem would make the advancing front methods much effective. I will investigate the skin triangulation using advancing front methods in Chapter 3 and attack the front collision problem by applying the recent results from computational topology studies.

Delaunay Mesh Generation. Delaunay meshing algorithms [28, 30, 42, 78, 87, 90, 58] utilize the *Delaunay triangulation* to generate meshes with provable guarantees on both shape quality and size. Delaunay mesh generators place mesh vertices to the boundary and interior of the domain followed by connecting them with the Delaunay triangulation. These two steps can be two separate phases but are usually integrated into a refinement procedure. That is, starting from the Delaunay triangulation of the boundary vertices, the algorithms maintain the Delaunay property of the triangulation while placing new mesh vertices at the circumcenters of some mesh elements until the whole domain boundaries appear in the mesh and the mesh quality satisfies the pre-set conditions.

Delaunay based approaches have become one of the most popular mesh generation methods because they not only offer nice theoretical guarantees on mesh quality and size but also perform excellently in practice. A key issue in applying the Delaunay based approach to mesh smooth surface is the efficiency. Since the complexity of the Delaunay triangulation of n surface samples can be $O(n^2)$ in the worst case [54], Delaunay surface meshing algorithms may be too slow. In this thesis, I improve the efficiency of this construction by combining the advancing front methods and Delaunay meshing in Chapter 4. Other challenges in Delaunay mesh generations include boundary recovering and sliver removal. I will further discuss these problems in Chapter 5.

Quadtree/Octree Methods. Meshing algorithms based on quadtrees (in two dimensions) and octrees (in three dimensions) use the *divide and conquer* strategy. An initial bounding cube (a square in two dimensions) is divided into eight congruent cubes followed by splitting these cubes recursively until each minimal cube intersects the domain in a simple way. Further splits are always performed to hold the balance condition, that is, no cube should be more than two times larger than its eight neighbors. The collection of all the cubes forms an octree decomposition of the domain. Then, the octree is wrapped and cut so that it conforms to the domain boundary. Finally, the cells in the octree are triangulated and forms the final meshes.

The quadtree/octree methods enjoy the same guaranteed quality as Delaunay meshing algorithm in two dimensions [14]. However, the mesh quality achieved in octree based surface meshing algorithms are usually bad because of the curvedness of the surface. Moreover, computing the intersection of a cube and the surface can be very costly, which decreases the efficiency of the meshing algorithm badly. As a result, I will not follow this method in my skin meshing studies.

To sum up, the framework of meshing techniques has been well established and fruitful results had been achieved in meshing the geometric domains such as polygons and polyhedra. Several challenges still reside in meshing the smooth surfaces. First, there should be provable bounds on the triangulation quality. Second, the output triangulation should be topological equivalent to the original surface. Finally, with the guarantees of mesh quality and topological correctness, the algorithm should be efficient and guaranteed to terminate. Overcoming these challenges in the study of skin meshing leads to the main contributions in this thesis.

1.4 Main Contributions

This thesis aims to develop efficient algorithms to generate quality surface and volumetric meshes for the skin surface. Since a quality surface triangulation is often essential to construct the volumetric mesh, the first goal of this thesis is to develop and implement surface triangulation algorithms for the skin satisfying the following requirements: (i) high efficiency; (ii) guaranteed quality; (iii) homeomorphic mesh; (iv) correctness and termination. I aim to triangulate the skin surface specified by thousands of spheres on a PC platform in a few minutes. At the same time, I will guarantee that the output triangulations have a lower bound on the minimal angle of the triangles in the mesh and are homeomorphic to the skin surfaces. Finally, I should demonstrate the correctness and termination of the triangulation algorithms. Based on the results of the surface meshing algorithms, the second goal of this work is to generate tetrahedral meshes for the volume enclosed by skin surfaces with quality guarantees, which means the shape of the tetrahedra in the mesh is close to that of a regular tetrahedron.

As a result, this thesis consists of two parts. The first part concentrates on the surface triangulation algorithms and the second part studies the tetrahedralization of the skin volume.

In the first part, I develop two skin triangulation algorithms, namely, the adaptive sweeping skin triangulation and Delaunay skin meshing using restricted union of balls. The first algorithm adapts to the advancing front method and applies the *Morse theory* to handle the front collision problem. A curvature adaptive scheme for the triangle size and quality control is designed in the algorithm to guarantee the mesh quality and the homeomorphism. In the process of sweeping meshes along the surface, I utilize the critical points of a height function defined on the surface to handle the front collision problem efficiently. Due to the robustness issue in the implementation raised by the noisy critical points, I present another skin surface triangulation algorithm capturing the advantages of both the advancing front and Delaunay mesh generation techniques. In this algorithm, I use the *restricted union of balls* to generate an ε-sampling of the skin surface. The sample points are generated incrementally and have a lower bound on the distance to their nearest neighbors. After each surface sample point is placed, the Delaunay triangulation of all the sample points is constructed with an incremental manner efficiently. A specified subset of the Delaunay triangulation, namely, the *restricted Delaunay triangulation*, forms a quality triangulation of the skin surface when the algorithm terminates.

In the second part, I introduce an algorithm to generate quality tetrahedral mesh for the volumes bounded by skin surfaces. By taking the advantages of the previous skin surface meshing results, the algorithm builds an initial Delaunay meshes bounded by the surface and applies the Delaunay refinement to improve the mesh quality afterwards. In particular, the algorithm inserts the circumcenters of bad shape tetrahedra with a priority parameterized by the value of the distance function defined by the surface. The algorithm achieves an upper bound on radius-edge ratio of the tetrahedral mesh after the refinement and all the slivers are removed in a post processing procedure. The algorithm terminates with guarantees on the tetrahedral quality and an accurate approximation of the original surface boundary.

The triangulation algorithms of this study will serve as a powerful tool for the study of the shapes and functions of molecules. First, the skin triangulation approximates the surface of a molecule and is useful for studying the shape features of the molecules. For example, the concave and convex features on the molecular surface, which are used to study of protein-docking problem, can be identified with skin meshes [86, 94]. Second, the good quality tetrahedral mesh of the skin body facilitates the numerical computations to approximate the electrostatic potentials of the proteins, which are essential to improve the reliability of the scoring function used in the protein-ligand docking programs [63]. Finally, the triangulation algorithms also provide new insights to the triangulation of other smooth surfaces and the domains with curved boundaries.

In the next chapter, I will introduce some geometric background and several important geometric properties of the skin surface, namely, the continuity of the curvature, the skin decomposition and the homeomorphic conditions, which are important for the development of skin triangulation algorithms.

Chapter 2

Geometric Background

The skin surface is a new paradigm of smooth surfaces based on the geometric notions of Voronoi diagram and Delaunay triangulation defined by a set of spheres. In particular, the skin triangulation algorithms developed in this thesis generate surface meshes that are the subsets of the Delaunay triangulation defined by the sample points on the surface. In this chapter, I will introduce these geometric backgrounds and develop relationships between the Voronoi diagram, Delaunay triangulation, alpha complex, skin surfaces and skin triangulations.

Section 2.1 reviews Voronoi diagram, Delaunay triangulation and their generalization to weighted point sets. A special subset of the weighted Delaunay triangulation, namely, the alpha complex, is introduced at the end of this section. With these notions as the foundations, I will introduce the definition and geometric properties of the skin surface in Section 2.2. Section 2.3 introduces the triangulation of skin surfaces and the homeomorphic conditions.

2.1 Voronoi and Delaunay Complexes

This section introduces the Voronoi diagram and its dual Delaunay triangulation. I begin with the terminology of simplicial complexes. Then, I review the definition of the Voronoi diagram and Delauany triangulation for a finite set of unweighted points and weighted ones. Finally, I introduce a special subset of the weighted Delaunay triangulation, namely, the alpha complex. Although these definitions apply to arbitrary fixed dimensions, I will focus on the three dimensional cases.

2.1.1 Simplicial Complexes

A *simplex* is the convex hull of a set of affinely independent points in $T \subset \mathbb{R}^d$, that is, $\sigma = \text{conv}(T)$. A set T is called *affinely independent* if every point $x \in T$ is not the affine combination of other points in T. The maximum number of affinely independent points in $\mathbb{R}^d$ is $d+1$. So in $\mathbb{R}^3$ we only have four types of simplices, that is, *vertices, edges, triangles and tetrahedron* when $\text{card}(T) = 1, 2, 3, 4$, in which $\text{card}(T)$ denotes the cardinality of the set T. The simplex σ is called a k-simplex and $k = \text{card}(T) - 1$ is the dimension of the simplex. The empty set is defined as a (-1)-simplex. Figure 2.1 shows the simplices of dimensions 0, 1, 2 ,3 from left to right. For any subset $S \subseteq T$, the simplex $\tau = \text{conv}(S)$ is called the *face* of σ, and the simplex σ is called the *coface* of τ.

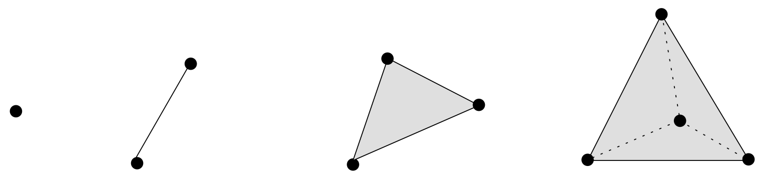


Figure 2.1: Four types of simplices.

A *simplicial complex* K is the collection of faces of a finite set of simplicies in which any two simplices are either disjoint or meeting in a common face. That is, the collection K satisfies the following two conditions,

1. if $\sigma \in K$ and τ is a face of σ, then $\tau \in K$, and

2. if σ, $\sigma' \in K$, then $\tau = \sigma \bigcap \sigma'$ is a face of both σ and σ'.

A *subcomplex* of K is a subset of K that is a simplical complex. The *underlying space* of K is the union of its simplices, denoted as $|K|$. The *star* of a 0 dimensional simplex $p \in K$ is the collection of its cofaces. See Figure 2.2 for an example. Figure (a) shows a simplicial complex that consists of 4 triangles, 9 edges and 6 vertices. One of its subcomplexes with 1 triangle, 4 edges and 4 vertices is showed in Figure (b). Figure (c) illustrates the underlying space of the simplical complex in (a). The six solid edges and 4 triangles in Figure (d) form the star of vertex p.

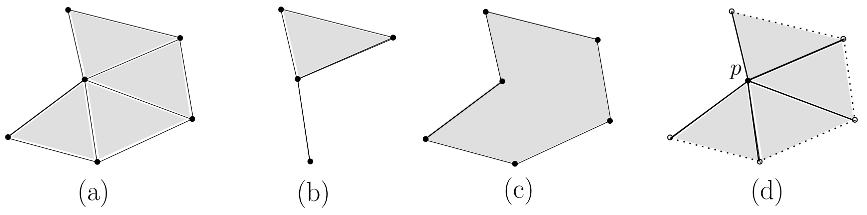


Figure 2.2: Examples of simplicial complexes. (a) a simplicial complex. (b) a subcomplex (c) the underlying space of (a). (d) the star of a vertex p.

2.1.2 Unweighted Voronoi and Delaunay Complex

Given a finite set of points $P = \{p_1, p_2, \cdots, p_n\} \subseteq \mathbb{R}^3$, the *Voronoi Diagram* of P is a subdivision of $\mathbb{R}^3$ in which each cell is the *Voronoi region* of a point p_i. The dual of the Voronoi diagram is called the *Delaunay triangulation* of P.

Voronoi Diagram. The *Voronoi region* of $p_i \in P$, denoted as ν_i, is the set of points in $\mathbb{R}^3$ that is closer to the point p_i than any other point in P, namely,

$$\nu_i = \{x \in \mathbb{R}^3 \mid \|x - p_i\| \leq \|x - p_j\|, \forall j \leq n\},$$

in which $\|x - p_i\|$ denote the Euclidean distance between two points x and p_i. Thus, ν_i is the intersection of $n-1$ half spaces defined by the bisector plane between p_i and

p_j, which is a convex polyhedron but possibly unbounded. Figure 2.3 (a) shows the Voronoi region of the red point in a point set consisting of ten points in $\mathbb{R}^3$.

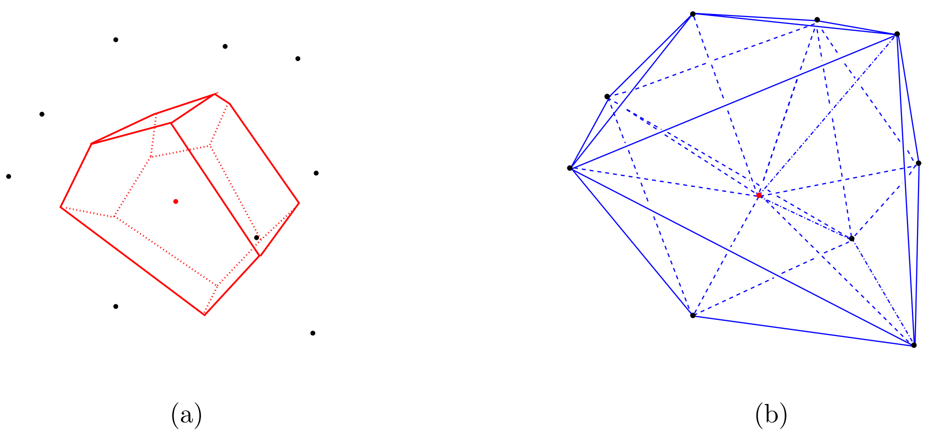


Figure 2.3: A Voronoi polyhedron and the Delaunay triangulation of 10 points in $\mathbb{R}^3$.

Voronoi regions may meet each other along a common portion of their boundary. The *Voronoi cell* of a subset $X \subseteq P$ is defined as the common intersections of the Voronoi regions, $\nu_X = \bigcap_{p_i \in X} \nu_i$. The *Voronoi diagram* of P is the collection of all the non-empty Voronoi cells, $V_P = \{\nu_X \mid \nu_X \neq \emptyset, X \subseteq P\}$. In $\mathbb{R}^3$, the Voronoi diagram consists of polyhedra, polygons, edges and vertices.

It is convenient to assume that the point set P satisfies the *general position* conditions, that is, there are no four points in a common plane and no five points on a common sphere. This assumption removes the degenerate cases in our discussions of Voronoi diagram and its dual. In practice, the assumption is not necessary since an arbitrary small perturbation can remove these degeneracies [48].

Delaunay Complex. The Delaunay complex of P, D_P, usually known as the *Delaunay triangulation*, consists of a collection of tetrahedra that decomposes the convex hull of P. See Figure 2.3 (b) for an example. Each tetrahedron τ, a 3-simplex, is the convex hull of four points whose Voronoi cell share a common Voronoi vertex. The faces of a tetrahedron τ are triangles, edges, and vertices, which are dual to Voronoi edges, Voronoi polygons and Voronoi polyhedra respectively.

The Delaunay complex D_P is a unique triangulation of P that has the *empty sphere* property. That is, the circumsphere of each tetrahedron in D_P does not enclose any point in P. It also implies that the diametral sphere of the Delaunay edges and the equatorial sphere of the Delaunay triangles are empty. As a result, the closest pair of points in P must be connected by an edge in the Delaunay triangulation. Another interesting result is that the minimum spanning tree of P is a subcomplex of the Delaunay Complex D_P [32].

Randomized Construction. The Delaunay triangulation can be constructed with a randomized incremental algorithm efficiently [51, 72]. This algorithm is based on the empty sphere property of the Delaunay tetrahedra. The basic idea of the algorithm is the following. Assuming that the Delaunay triangulation D_i of the first i points in P is already constructed. Add the $(i + 1)$-th point into the triangulation D_i and restore the Delaunayhood by edge flipping, this results in D_{i+1}. Repeat this process until $i = n$. Each edge flip replaces two tetrahedra with three other tetrahedra or vice versa. Figure 2.4 illustrates an example of the flipping.

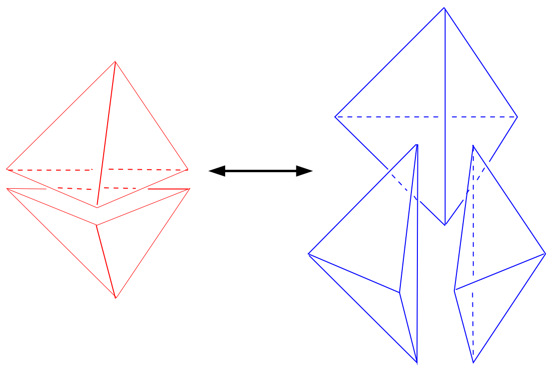


Figure 2.4: An edge flip for computing the Delaunay triangulation in $\mathbb{R}^3$.

A crucial step in the algorithm is the point location, which occurs when a new point is added into the triangulation. A directed acyclic graph (DAG) with the history of all performed flips are used to speed up the point location. The expected running time of the algorithm is $O(n \log n)$ in $\mathbb{R}^3$. However, it could run into $O(n^2)$ in the worst case but rarely occurred in practice [54]. The ability to permit efficient algorithms makes the Delaunay triangulation very popular in the applications such as mesh generation, geography and computer graphics [83, 90, 98].

2.1.3 Weighted Case

So far, the points are unweighted, and I will generalize the concepts to weighted point set in this section. A weighted point $b_i \in \mathbb{R}^3 \times \mathbb{R}$ is denoted as $b_i = (z_i, w_i)$, in which z_i is the position of b_i and w_i is its weight. We can also view b_i as a sphere or a ball with center z_i and radius $\sqrt{w_i}$. The weight w_i can be negative, and it represents an imaginary sphere. A finite set of n weighted points is denoted as $B = \{b_i = (z_i, w_i) \in \mathbb{R}^3 \times \mathbb{R} \mid i = 1..n\}$. The Voronoi diagram of B is the generalization of the unweighted Voronoi diagram by replacing the Euclidean distance with weighted distances.

Weighted Distance. For a point $x \in \mathbb{R}^3$, the *weighted distance* from x to the weighted point b_i defined by

$$\pi(x, b_i) = \|x - z_i\|^2 - w_i.$$

Geometrically, the weighted distance can be explained as the length of the tangent line segment xt of sphere b_i passing through x, as shown in Figure 2.5. We have $\pi(x, b_i) = 0$ when x is on the boundary of b_i and $\pi(x, b_i) < 0$ when x is in the interior of b_i. The set of points with equal weighted distance from two spheres in B is a hyperplane. In particular, if the two spheres intersect with each other, the hyperplane passes through the intersection circle. Figure 2.5 illustrates the ideas with

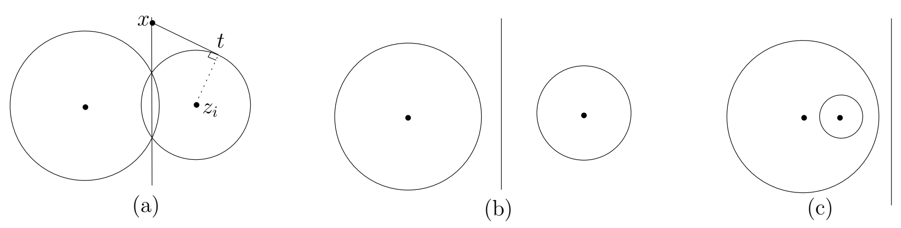


Figure 2.5: The weighted distance and the bisector of two circles.

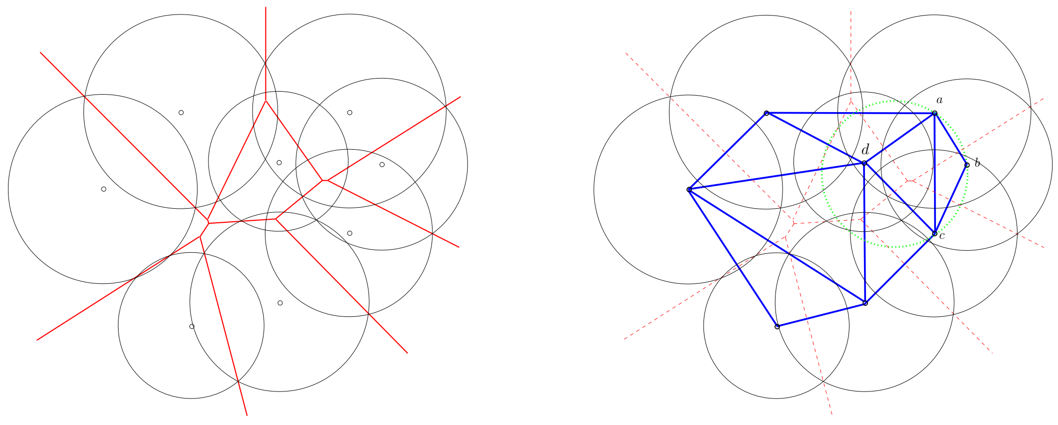
the three possible configurations of two circles.

Weighted Voronoi and Delaunay Complex. The Voronoi region ν_i of the weighted point b_i is defined as the set of the points in $\mathbb{R}^3$ with smaller weighted distances to b_i than any other weighted points in B, namely, $\nu_i = \{x \in \mathbb{R}^3 \mid \pi(x, b_i) \leq \pi(x, b_j), \forall j \leq n \quad \text{and} \quad j \neq i\}$. Similar to the unweighted case, the Voronoi region ν_i is a convex polyhedron (possibly unbounded) that is the intersection of $n-1$ half spaces defined by the inequalities. Note that there are situations that $\nu_i = \emptyset$. The weighted point b_i is called *redundant* if $\nu_i = \emptyset$. The Voronoi regions overlap each other with polygons that are the the Voronoi cell for a subset $X \subseteq B$, namely, $\nu_X = \bigcap \nu_i, b_i \in X$. The *weighted Voronoi diagram* is the collection of all the non-empty Voronoi cells, $V_B = \{\nu_X \mid \nu_X \neq \emptyset, X \subseteq B\}$. Figure 2.6 (a) illustrates a weighted Voronoi diagram defined by 8 circles in $\mathbb{R}^2$.

For a subset $X \subset B$ with a non-empty Voronoi cell, define δ_X as the convex hull of the centers of the weighted points in X, $\delta_X = \text{conv}(\{z_i \mid b_i \in X\})$. The weighted Delaunay triangulation D_B of B is

$$D_B = \{\delta_X \mid \nu_X \in V_B\}.$$

Figure 2.6 (b) shows the weighted Delaunay triangulation of 8 circles in $\mathbb{R}^2$. The weighted Delaunay triangulation is a triangulation of the convex hull of the centers of all non-redundant weighted points. However, it may not be the Delaunay triangula-



(a) (b)

Figure 2.6: The weighted Voronoi diagram and Delaunay triangulation of 8 circles in $\mathbb{R}^2$.

tion of the centers. For example, the triangle *abc* in Figure 2.6 (b) is not a Delaunay triangle because its circumcircle encloses another vertex d.

In $\mathbb{R}^3$, we assume the general position such that no more than 4 weighted points with Voronoi cells sharing a common Voronoi vertex. Also, each Voronoi edge belongs to exactly 3 Voronoi regions. As a result, the Voronoi cells are vertices, edges, polygons and polyhedra respectively when $\text{card}(X) = 4, 3, 2, 1$. Under the general position assumption, D_B is a simplicial complex. Each tetrahedron $\tau \in D_B$ and its faces, namely, triangles, edges, and vertices are dual to a Voronoi vertex, edges, polygons and polyhedra respectively. This dual relationship is illustrated in Figure 2.7.

Since the empty sphere property for unweighted Delaunay triangulation does not hold for the weighted cases, we define the *orthogonal sphere* to generalize the empty sphere criteria for the weighted Delaunay triangulation.

Orthogonal Spheres. For two spheres b_i, b_j, the weighted distance between them is defined as $\pi(b_i, b_j) = \|x - z_i\|^2 - w_i - w_j$. The sphere b_i is *orthogonal* to the sphere b_j if and only if their weighted distance $\pi(b_i, b_j) = 0$. Geometrically, the spheres b_i and b_j

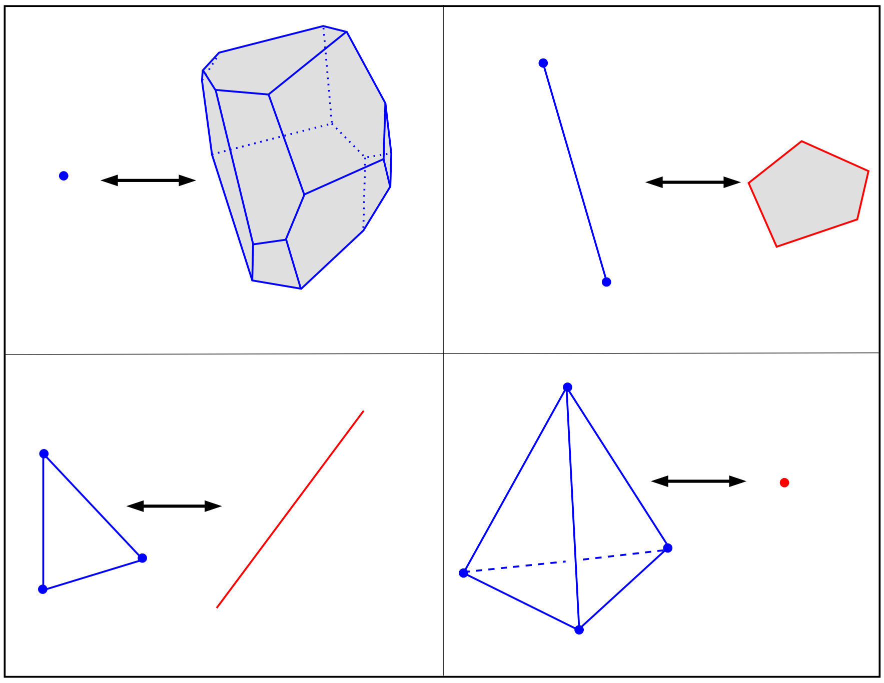


Figure 2.7: The dual relationship between Delaunay simplices and Voronoi Cells.

intersect at a right angle. Two spheres are called *further than orthogonal* when their weighted distance is positive, and *closer than orthogonal* when the distance becomes negative.

For a tetrahedron $\tau \in D_B$, let z_τ be the dual Voronoi vertex and z_τ has a equal weighted distance r from four balls b_1, b_2, b_3, b_4 whose centers are exactly the vertices of τ. We define the sphere $b = (z_\tau, r)$ as the *orthogonal sphere* of the Delaunay tetrahedron τ because b is orthogonal to the spheres b_1, b_2, b_3 and b_4. At the same time, the sphere b is further than orthogonal from all other spheres in B and we call the orthogonal sphere *empty.* This property is used to generalize the empty sphere property for a set of weighted points. The weighted Delaunay triangulation D_B consists of all the tetrahedra with vertices z_i, z_j, z_k, z_l such that the orthogonal sphere of b_i, b_j, b_k, b_l is empty. Note that it is also true for the unweighted case if we consider each unweighted point as a weighted point with zero weight.

As a result, we can apply the randomized incremental algorithm for computing the Delaunay triangulation of (unweighted) point set P to compute the weighted Delaunay triangulation by generalizing the empty sphere criteria. Edelsbrunner and Shah [51] described such an efficient algorithm to compute weighted Delaunay triangulations. In general, the weighted Delaunay triangulation of B is different from the Delaunay triangulation of the unweighted point set $Z = \{z_i \mid \forall b_i = (z_i, w_i) \in B\}$ except the case that all the spheres in B have same weights.

The weighted Delaunay triangulation has a number of applications in mesh generation and surface reconstructions [27, 28, 57]. I will use the weighted Delaunay triangulation to remove the slivers in the Delaunay tetrahedralization in Chapter 5. In the next section, I will investigate a subset of the weighted Delaunay triangulation that characterizes the shape of a union of the balls.

2.1.4 Alpha Complex

The *α-complex* is a subcomplex of the weighted Delaunay triangulation D_B parameterized by a real value α. When $\alpha = 0$, the α-complex is usually called the *dual complex* of the *union of the balls* in B, which is defined as $\bigcup B = \{x \in \mathbb{R}^3 \mid \pi(x, b_i) \leq 0, \forall i \in [1 \cdots n]\}$.

Dual Complex. The union of balls in B covers only a portion of the Voronoi cells in V_B and each Voronoi cell is dual to a Delaunay simplex in D_B. The *dual complex* of $\bigcup B$, denoted as K_B, is defined as the collection of Delaunay simplices whose dual Voronoi cells have non-empty intersection with the union of balls, that is,

$$K_B = \left\{\delta_X \in D_B \mid \bigcup B \cap \nu_X \neq \emptyset, X \subseteq B\right\}.$$

Figure 2.8 shows the dual complex of 8 disks in two dimensions. The dashed lines are the Voronoi edges. The solid lines and triangles form the dual complex. The dual

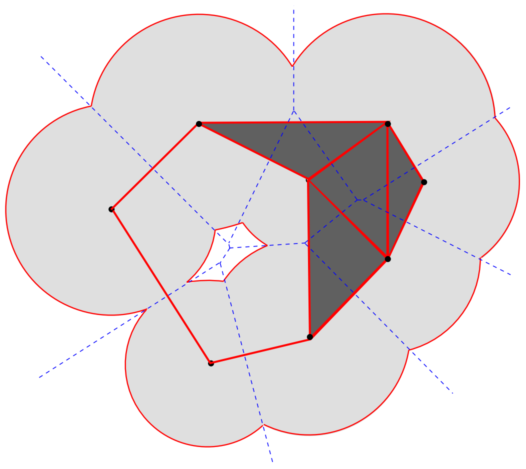


Figure 2.8: The dual complex of 8 disks.

complex K_B has the same topology with the union of balls in B. More precisely, the dual complex K_B is homotopy equivalent to $\bigcup B$ and there is a deformation retraction from $\bigcup B$ to K_B. We refer to [36] for a complete list of the properties of the dual complex. Since the dual complex is a simplicial complex and there are efficient combinatorial algorithms to compute its topological properties such as Betti numbers [34], we can investigate the topological properties of a union of balls by studying its dual complex.

Alpha Complex. We can grow the balls in B with a parameter α and generate a new set of balls. In particular, we choose the growth model that keeps the weighted Voronoi diagram V_B unchanged. That is, we define a new set of balls for the control value α as the following,

$$B(\alpha) = \{b_i(\alpha) = (z_i, w_i + \alpha^2), b_i \in B\}.$$

The dual complex of the union of balls in $B(\alpha)$ is referred as the *α-complex*, namely,

$$K_\alpha = \Big\{\delta_X \mid \bigcup B(\alpha) \cap \nu_X \neq \emptyset, X \subseteq B\Big\}.$$

The alpha complex is changed as the value of α varies. When the α value is negative and small enough, all the balls in $B(\alpha)$ are imaginary and its α-complex is empty. As we increase α, some simplices in D_B enter the alpha complex $K(\alpha)$ until the α-complex is equal to the Delaunay triangulation D_B. Figure 2.9 shows the α-complex of 8 disks at three moments.

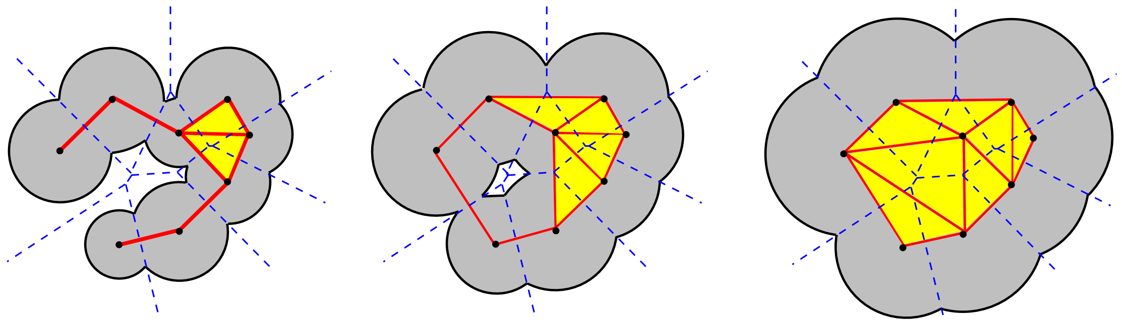


Figure 2.9: Uniformly growing disks and their α-complexes.

As the control value α grows, the changing of the α-complex is in a way of gaining new simplices. For each simplex $\delta_X \in D_B$, its *birth time* ζ_X is defined as the corresponding α value such that the union of balls in $B(\alpha)$ just touches the Voronoi cell ν_X. And δ_X will be in the α-complex K_α for all $\alpha \geq \zeta_X$. For a 3-simplex, that is, a tetrahedron, the birth time is exactly the radius of its *orthosphere*. For a simplex δ_X with dimension less than 3, we define the *orthosphere* of δ_X as the smallest sphere b that orthogonal to all spheres in X, $b = (z_\delta, r)$. The point z_δ is called the *center* of the simplex and it is the intersection of the affine space of δ_X and ν_X. The radius of the orthosphere r is the weighted distance from z_δ to either sphere in X.

After a Delaunay simplex enters the α-complex, the topology of the α-complex changes in a specific way depending on the dimension of the simplex. That is, when a 0-simplex enters the α-complex, a new component is formed by the vertex itself; when a 1-simplex enters the α-complex, two components or two portions of one component are connected by the edge; when a 2-simplex appears in the α-complex, the triangle either splits a void, which is a component of the complementary part of the α-complex in $\mathbb{R}^3$, or closes a tunnel between two portions of the same void; when a 3-simplex appears, it fills a void.

As a result, we can compute the time and types of the topological changes of the α-complex deterministically. Since the dual complex has the same topological type as the union of balls, we can investigate the topological changes of the union of balls by studying the α-complex. Furthermore, the topology of the space bounded by the *skin surface* changes in the same way as the α-complex. I will first introduce the definition of the skin surface in the next section.

2.2 Skin Surfaces

The skin surface, denoted as F_B, specified by a finite set of spheres B is a closed C^1-continuous surface in $\mathbb{R}^3$. It consists of one or more disjoint components and each one is free of self-intersections and intersections with other components. Intuitively, the skin surface is geometrically similar to the boundary of the union of balls but with a smooth appearance by blending the spheres with quadratic patches. Figure 2.10 illustrates the union of a set of spheres that forms a torus and the corresponding skin surface.

In this section, I first introduce the sphere algebra and define the skin surface as the envelope of the convex combination of the spheres after shrinking. Then, I describe the *mixed complex* to decompose the skin surface F_B into a finite collection of quadratic patches. Finally, I introduce the key properties of the skin surface, namely, the complementarity property, the curvature variation of the skin surface and its capability of deformation.

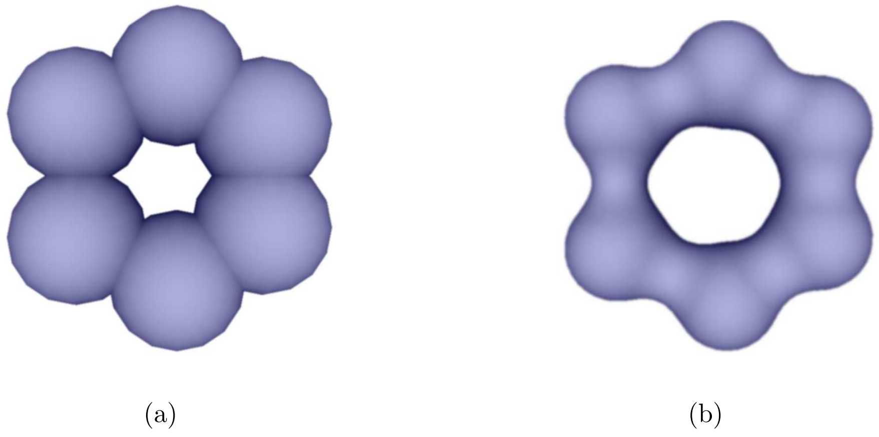


Figure 2.10: The union of spheres and the skin surface model of a torus.

2.2.1 Skin Definition

Recall that we can consider a sphere $b_i \in B$ as the zero-set of the *weighted distance function* $\pi_i(x) = \|x - z_i\|^2 - w_i$. Following the addition and the scalar multiplication operations of the weighted distance functions $\pi_i(x)$, we can define the addition of two spheres b_i, b_j, and the scalar multiplication of b_i and a real number c as

$$\begin{aligned} (z_i, w_i) + (z_j, w_j) &= (z_i + z_j, w_i + w_j + 2\langle z_i, z_j \rangle), \quad \text{and} \\ c \cdot (z_i, w_i) &= (c \cdot z_i, c \cdot (w_i - (1-c)\|z_i\|^2)). \end{aligned}$$

The notation $\langle z_i, z_j \rangle$ is the dot product of vectors z_i and z_j. Using these two operations, the *affine hull* and *convex hull* of $X \subseteq B$ can be defined as

$$\begin{aligned} \text{aff}(X) &= \left\{ \sum_{b_i \in X} \lambda_i b_i \middle| \sum_i \lambda_i = 1, \right\}, \quad \text{and} \\ \text{conv}(X) &= \left\{ \sum_{b_i \in X} \lambda_i b_i \middle| \sum_i \lambda_i = 1, \forall \lambda_i \geqslant 0 \right\}. \end{aligned}$$

The affine hull and convex hull of X consist of an infinite family of spheres. For any sphere $b_j \in \text{aff}(X)$, we can deduce the expression of the center and radius of b_j as

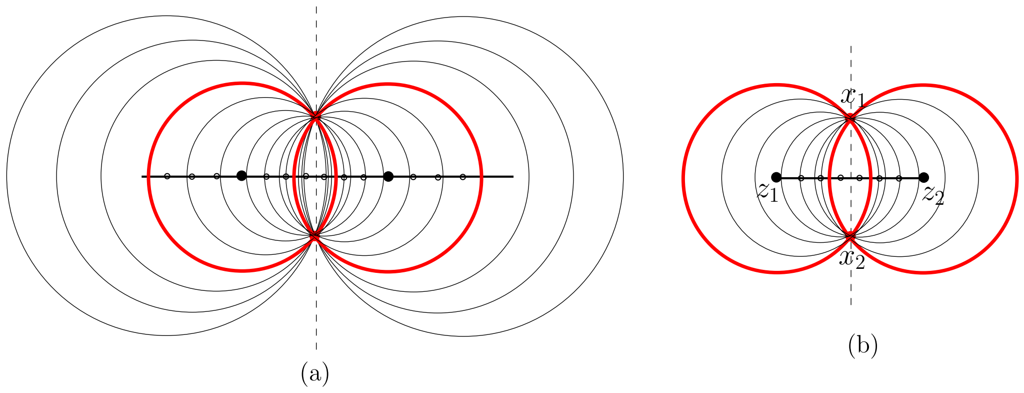


Figure 2.11: The affine hull and convex hull of two circles.

following,

$$
\begin{aligned}
z_j &= \sum_i \lambda_i z_i, \\
w_j &= \sum_i \lambda_i w_i + \Big\|\sum_i \lambda_i z_i\Big\|^2 - \sum_i \lambda_i \|z_i\|^2.
\end{aligned}
$$

That is, the center of b_j is on the affine hull of the centers of the spheres in B. For example, if $X = \{b_1, b_2\}$ and b_1, b_2 intersect each other with a common circle in $\mathbb{R}^3$, the affine hull contains all the spheres that pass through the circle and have their centers lying on the line passing through z_1 and z_2. Since the family of spheres are difficult to draw, I illustrate the affine hull and convex hull of two circles in two dimensions in Figure 2.11. We can also consider it as the cross section of three dimensional cases.

In order to define the skin surface, we need one more operation. We define the shrinking operation of a sphere b_i as $\sqrt{b_i} = (z_i, \frac{w_i}{2})$, and for a set of spheres X, $\sqrt{X} = \{\sqrt{b_i} \mid b_i \in B\}$. Figure 2.12 shows the shrunk affine hull and convex hull of two circles.

We are interested in the envelope of the family of spheres in the affine hull and convex hull of X after shrinking. In $\mathbb{R}^3$, Edelsbrunner [38] proved that the envelope of aff($\{b_i, b_j\}$) is a hyperboloid and the envelope of conv($\{b_i, b_j\}$) is a closed smooth

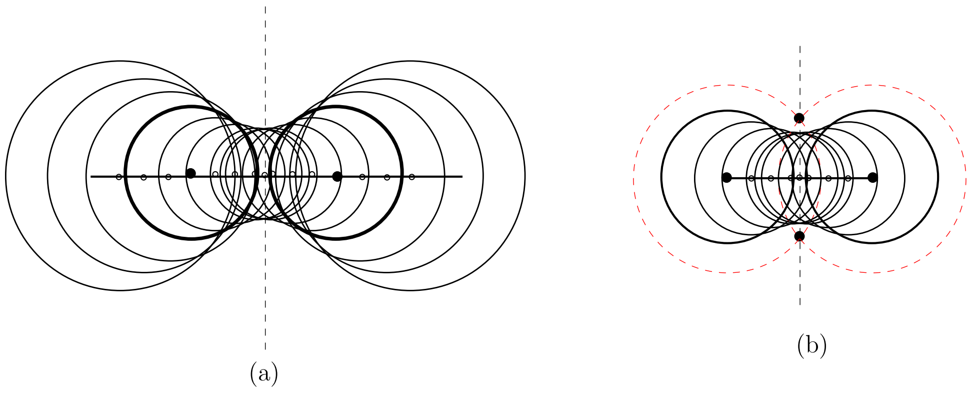


Figure 2.12: The shrunk affine hull and convex hull of two circles.

surface consisting of two spherical patches joined by a portion of a hyperboloid, as shown in Figure 2.13.

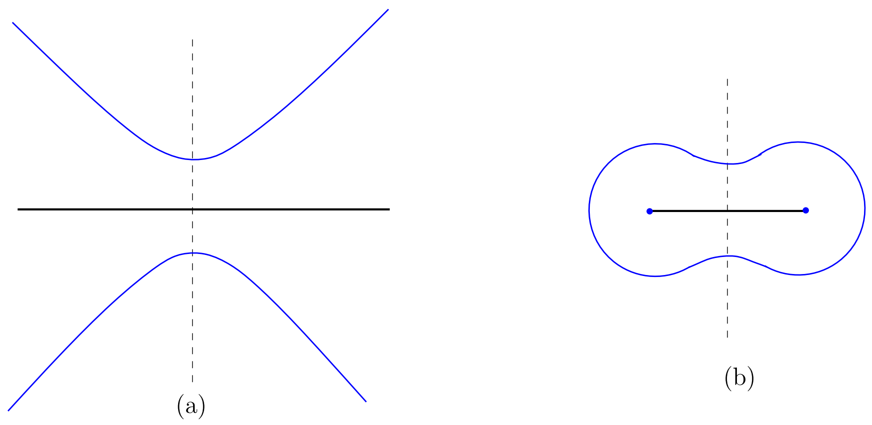


Figure 2.13: The envelope of the shrunk affine hull and convex hull of two circles.

For a general set of spheres B, the skin surface, F_B, is defined as the envelope of the family of infinite spheres in the convex hull of B after shrinking, namely,

$$F_B \quad = \quad \text{env}\left(\sqrt{\text{conv}(B)}\right).$$

The envelope is the boundary of the union of balls in the shrunk convex hull of B. The union of balls, $\bigcup\sqrt{\text{conv}(B)}$, is called the *body* of the skin F_B. It is difficult to illustrate the shape of a general skin surface directly as what we did in Figure 2.13 for two circles. Instead, I use an indirect way to illustrate the shape of the skin surface in the next section. First, I decompose the skin surface to a collection of patches specified by up to four spheres. Then, I show that each patch is a portion of either a sphere or a hyperboloid.

2.2.2 Skin Patches

I introduce the *mixed complex* that decompose the skin surface into a collection of simple pieces. Recall that the *Voronoi complex* V_B of B and its dual *Delaunay complex* as D_B establish a corresponding relationship between each Delaunay cell and its dual Voronoi cell of a subset $X \subseteq B$. When X has cardinality 1 to 4, the Delaunay cells are vertices, edges, triangles and tetrahedra in $\mathbb{R}^3$ respectively. Then, the corresponding Voronoi cells are polyhedra, polygons, edges and vertices respectively.

The *mixed cell*, μ_X, of X is defined as the Minkowski sum of ν_X and δ_X shrunk by half, that is,

$$\mu_X \quad = (\nu_X + \delta_X)/2.$$

The mixed cell, μ_X, is always a polyhedron. Figure 2.14 illustrates the four cases of the mixed cell in three dimensions. The collection of mixed cells partitions the space

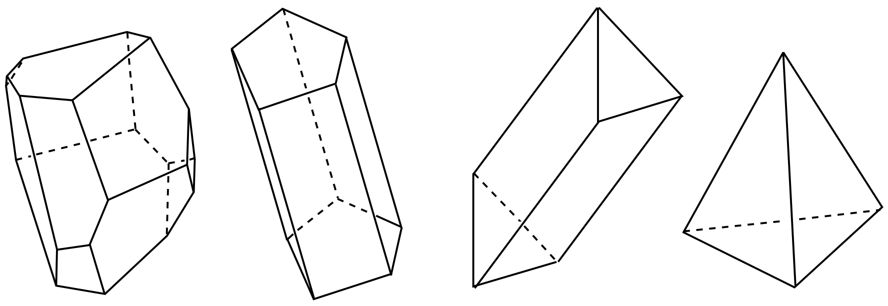


Figure 2.14: Four different mixed cells with dimension from 1 to 4.

in $\mathbb{R}^3$, which is called the mixed complex, M_B, of B. We define the *center* of μ_X as the point $z_X = \text{aff}(\nu_X) \bigcap \text{aff}(\delta_X)$, in which $\text{aff}(Y)$ is the affine hull of Y.

We consider the portion of the skin surface F_B clipped within each mixed cell μ_X. Edelsbrunner [38] proved that the intersection of the skin surface F_B and a mixed cell μ_X is same as the intersection of μ_X with the envelope of the shrunk affine hull of X, namely,

$$F_B \bigcap \mu_X = \text{env}\Big(\sqrt{\text{aff(X)}}\Big) \bigcap \mu_X.$$

In other words, the skin surface within each mixed cell is determined by the weighted points in X. Under the general positions assumption, we have only four different cardinalities of X, namely, $\text{card}(X) = 1, 2, 3$ or 4.

When $\text{card}(X) = 1$, the affine hull of $X = \{b_i\}$ is the sphere b_i itself. As a result, the skin patch in μ_X is simply the part of the sphere with center at z_i and radius $\sqrt{\frac{w_i}{2}}$ clipped by the faces of μ_X.

When $\text{card}(X) = 2$, the envelope of the shrunk affine hull of $X = \{b_i, b_j\}$ is a hyperboloid. The idea is illustrated in Figure 2.12. If we translate and rotate the mixed cell such that its center z_X is the origin, the hyperboloid can be expressed in a standard form, that is,

$$x_1^2 + x_2^2 - x_3^2 = \pm R^2.$$

in which R is the minimum distance from the center z_X to the hyperboloid and equal to the absolute value of the radius of the orthosphere of δ_X divided by $\sqrt{2}$. The sign in the right side of the equation is determined by the sign of the orthosphere radius. In the case of the orthosphere has a positive radius, the sign in the equation is minus and it indicates the hyperboloid is a one-sheeted hyperboloid. If the radius is negative, the envelope is a two-sheeted hyperboloid. If the radius is equal to zero, the envelope is a double cone, which is considered as the degenerate case. Figure 2.15

illustrates the hyperboloids. As a result, the skin patches within the mixed cell μ_X of dimension 1 are hyperboloid patches of 1 sheeted or 2 two sheeted.

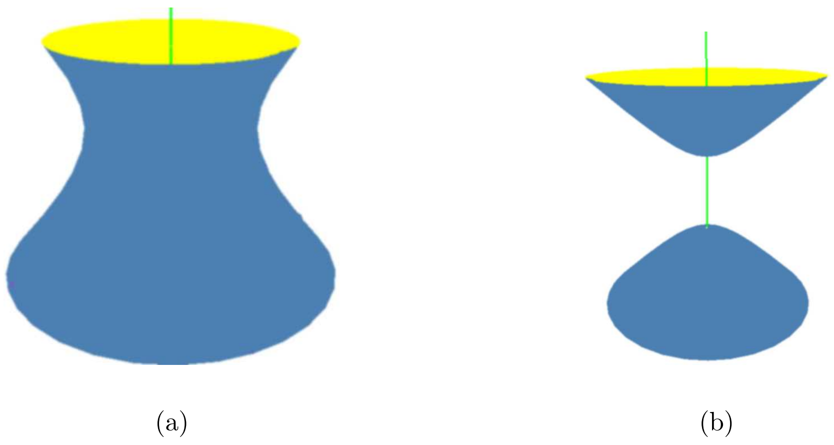


Figure 2.15: Examples of one sheeted hyperboloid and two sheeted hyperboloid.

For the case of $\text{card}(X) = 3$, we claim that the envelope of affine hull of X after shrinking is a hyperboloid as well. In order to explain this, we need the help of the orthospheres b_j, b_k of the coface tetrahedra next to the Delaunay triangle δ_X. Since the orthosphere b_j and b_k are orthogonal to every sphere in X, the sphere b_j and b_k are orthogonal to every sphere in the affine hull of X. By symmetry, every sphere $b_i \in \sqrt{\text{aff(X)}}$ is orthogonal to b_j and b_k, and every sphere in their affine hull as well. If we apply the shrinking operation to the spheres in $\sqrt{\text{aff(X)}}$ and $\sqrt{\text{aff}(\{b_j, b_k\})}$, we can find out that they share a common envelope, which is illustrated in Figure 2.16 using the two dimensional case. The reason is that two shrunk orthogonal spheres will touch each other if and only if they have the same radius, and they are disjoint in all other cases. See Figure 2.17 for the examples in two dimensions. As a result, the skin patch for X with cardinality 3 is also a hyperboloid patch.

Finally, when $\text{card}(X) = 4$, the orthogonal set of the four spheres in X is its unique orthosphere. As a result, the skin patch is the shrunk orthosphere clipped by the four faces of the mixed cells.

To sum up, we have two types of surface patches inside each μ_X depending on

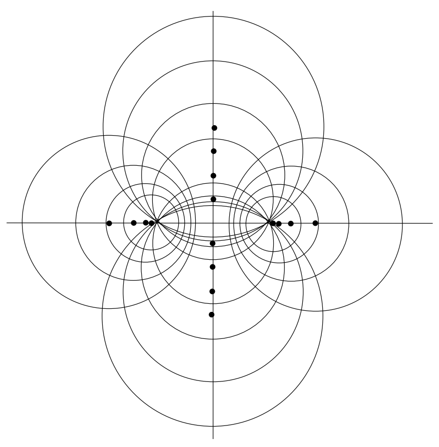
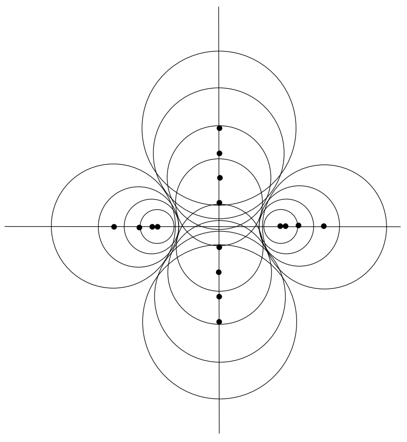


Figure 2.16: The orthogonal set and its shrunk result.

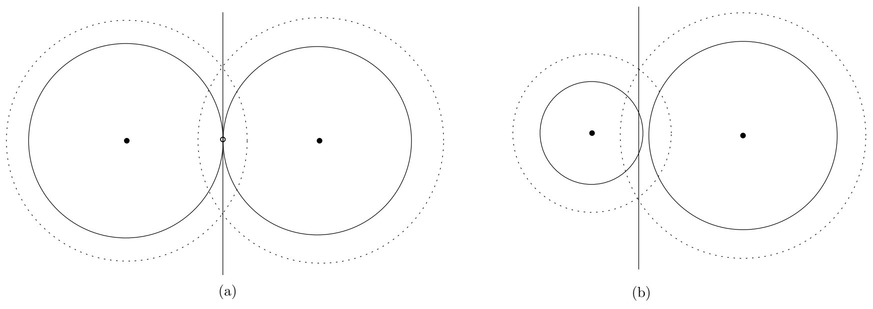


Figure 2.17: Two orthogonal spheres.

the cardinalities of X. If $\mathrm{card}(X) = 1$ or 4, there is a sphere patch clipped within the mixed cell μ_X. If $\mathrm{card}(X) = 2$ or 3, there is a hyperboloid patch clipped within μ_X. The center of μ_X, z_X, is the center of the corresponding sphere or the apex of the corresponding hyperboloid. These patches join each other smoothly at the intersection of two mixed cells and form a closed smooth surface. Figure 2.18 illustrates a skin surface and its patches in four different types of mixed cells.

2.2.3 Geometric Properties

The skin surface has a number of elegant properties [38]. Among them, I will introduce the complementarity, the curvature variation property, and the deformation under the

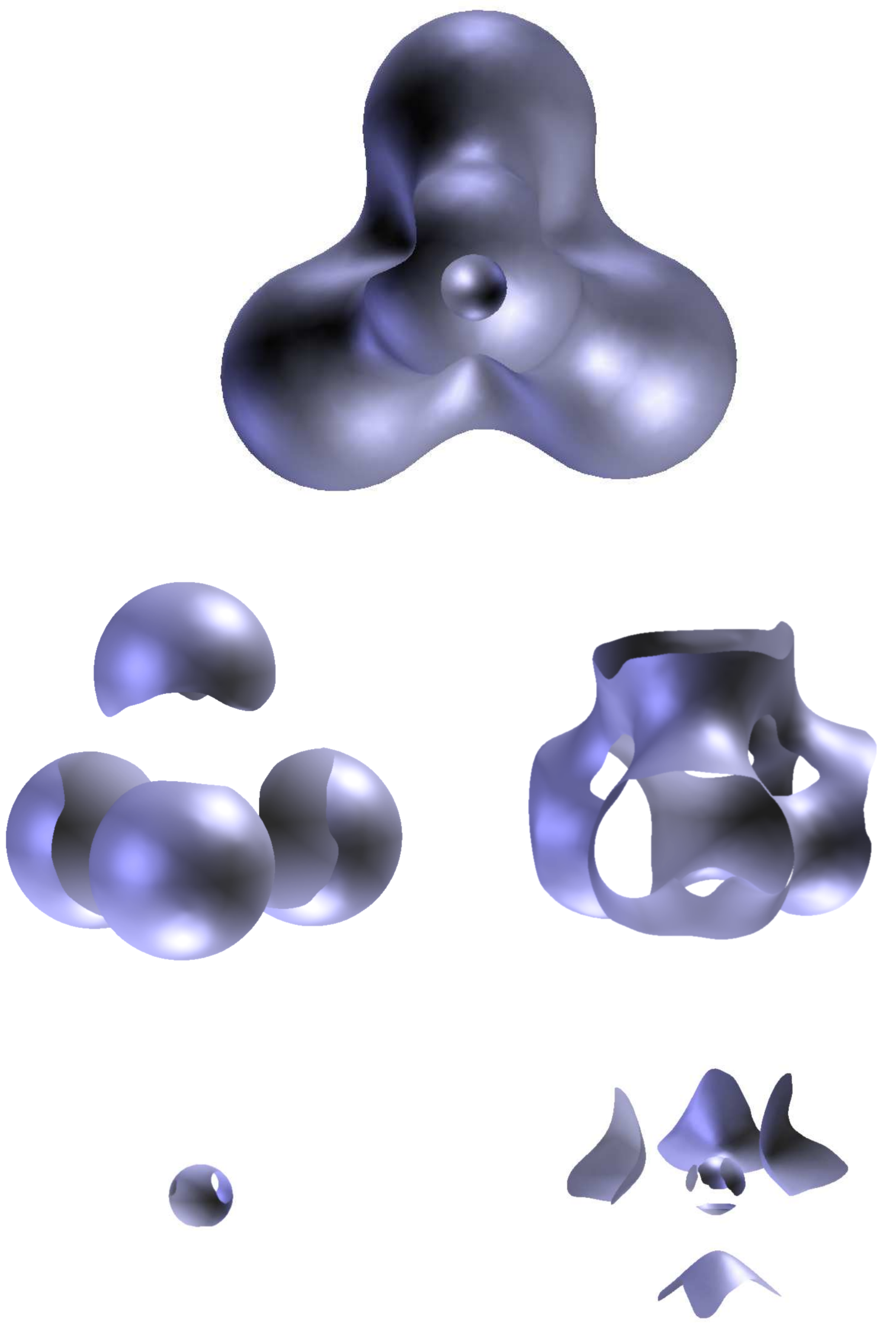


Figure 2.18: The skin patches clipped within the mixed cells.

growing model in this section to facilitate our discussions about the skin triangulation algorithms in the later chapters.

Complementarity. The complementarity of a skin surface F_B tells that we can find another collection of spheres that specify the same skin surface of B but with reverse normal direction and a complementary body. Edelsbrunner [38] shows that the orthogonal set of of B, $B^\perp = \{b_X^\perp = (z_X^\perp, w_X^\perp) \mid \delta_X \in D_B\}$, shares the same envelope with B, where $b_X^\perp$ is the orthogonal sphere of the simplex δ_X. The union of the body bounded by the skin of B and $B^\perp$ cover the whole three dimensional space. That is,

$$\begin{aligned}\text{body}(B)\bigcap\text{body}(B^\perp) &= \text{skin}(B)\\ &= \text{skin}(B^\perp),\\ \text{body}(B)\bigcup\text{body}(B^\perp) &= \mathbb{R}^3.\end{aligned}$$

In other words, the skin surface F_B is the envelope of two families of spheres, $\sqrt{\text{conv}(B)}$ and $\sqrt{\text{conv}(B^\perp)}$, one inside and the other outside the surface. As a result, for every point x on the skin surface F_B, there are two unique spheres b_x and $b_x^\perp$ that pass through x and externally tangent each other. These two spheres have same radius and their separate plane is the tangent plane at x. We refer to b_x and $b_x^\perp$ as the *sandwiching spheres* at x because they squeeze the surface in the local neighborhood around x. Figure 2.19 illustrates the ideas in two dimensions.

The radius of the sandwiching sphere varies continuously over the skin surface, which certifies the continuity of the tangent of $x \in F_B$. In addition, the maximum curvature of the skin surface is continuous and varies slowly on the surface.

Curvature Variation. The curvature of a surface at a point x is measured by using the curvature of the curve at the cross-section of the surface with a plane passing the

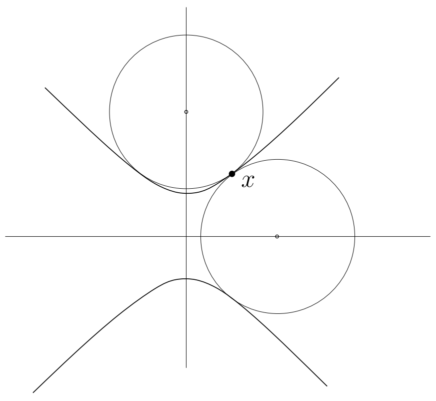


Figure 2.19: The sandwich spheres of a point on the skin surface.

normal of x. More precisely, given a surface $\mathbb{M}$, a point x on $\mathbb{M}$ with the tangent vector t_x, the normal curvature of $\mathbb{M}$ at x is that of the curve on $\mathbb{M}$ through x in the direction t_x. There is a circle of tangent vectors at the point x and each one corresponds to a normal curvature. The maximum and minimum normal curvatures at the point x on a surface are called the principle (normal) curvature [18]. The principle curvatures measure how much the surface is curving. I will investigate the changing of the maximum principle curvature of the skin surface and use it control the surface mesh size in our later studies.

Denote the maximum principle curvature at $x \in F_B$ as $\kappa(x)$. The reciprocal $1/\kappa(x)$ is defined as the *local length scale* at x, denoted as $\varrho(x)$. Since the skin surface can be decomposed into sphere patches and hyperboloid patches, we can compute $\kappa(x)$ analytically for any point x on a skin surface F_B. For a sphere, the maximum curvature is one over its radius. In the situation of a hyperboloid of revolution, the maximum normal curvature is one over the radius of its sandwiching sphere, which is equal to the distance of x from its apex. That is, for any point x on the skin surface F_B, the local length scale at x, $\varrho(x)$, is the distance from x to the center of the mixed cell. This means both the maximum curvature and the local length scale for each $x \in F_B$ can be determined individually within its mixed cell.

The maximum curvature varies slowly on the skin surface. For two points $x, y \in$

F_B, this property can be expressed in the Lipschitz condition [23],

$$|\varrho(x) - \varrho(y)| \leq \|x - y\|.$$

This property leads to the possibility of adaptive homeomorphic triangulation. I will use it to determine the step size of our algorithm in Chapter 3 and Chapter 4.

Growing Skin. Recall the growth model of B that keeps the weighted Delaunay triangulation D_B unchanged by introducing a parameter α, and defines the set of spheres $B(\alpha) = \{b_i(\alpha) = (z_i, w_i + \alpha) \in \mathbb{R}^3 \times \mathbb{R} \mid i = 1..n\}$. For each α-value, we have a skin surface $F_B(\alpha)$ that shares the same mixed cells with F_B. As we increase the α from $-\infty$ to ∞, the skin surface $F_B(\alpha)$ grows continuously and its topology changes at the center of each mixed cell [38].

Since the skin body of $F_B(\alpha)$ is homotopic to the underlying space of its dual complex $|K_B|$, the topological changes of the skin surface $F_B(\alpha)$ are corresponding to the topological changes of the alpha complex we introduced in Section 2.1.4. In particular, the changes of the topology depend on the cardinality of $X \subseteq B$. We have four cases for the cardinality of X, namely, $\text{card}(X) = 1, 2, 3, 4$. In the case of $\text{card}(X) = 1$, the skin surface in the mixed cell μ_X changes from empty to a sphere when the radius of $b_i(\alpha)$ become positive. When $\text{card}(X) = 2$, the skin changes from a 2-sheet hyperboloid to a 1-sheet hyperboloid. Symmetrically, in the case of $\text{card}(X) = 3$, the skin changes from a 1-sheet hyperboloid to a 2-sheet hyperboloid. When $\text{card}(X) = 4$, the skin changes from a sphere to empty. Figure 2.20 illustrates 5 snap-shots of the growing skin surface specified by four spheres.

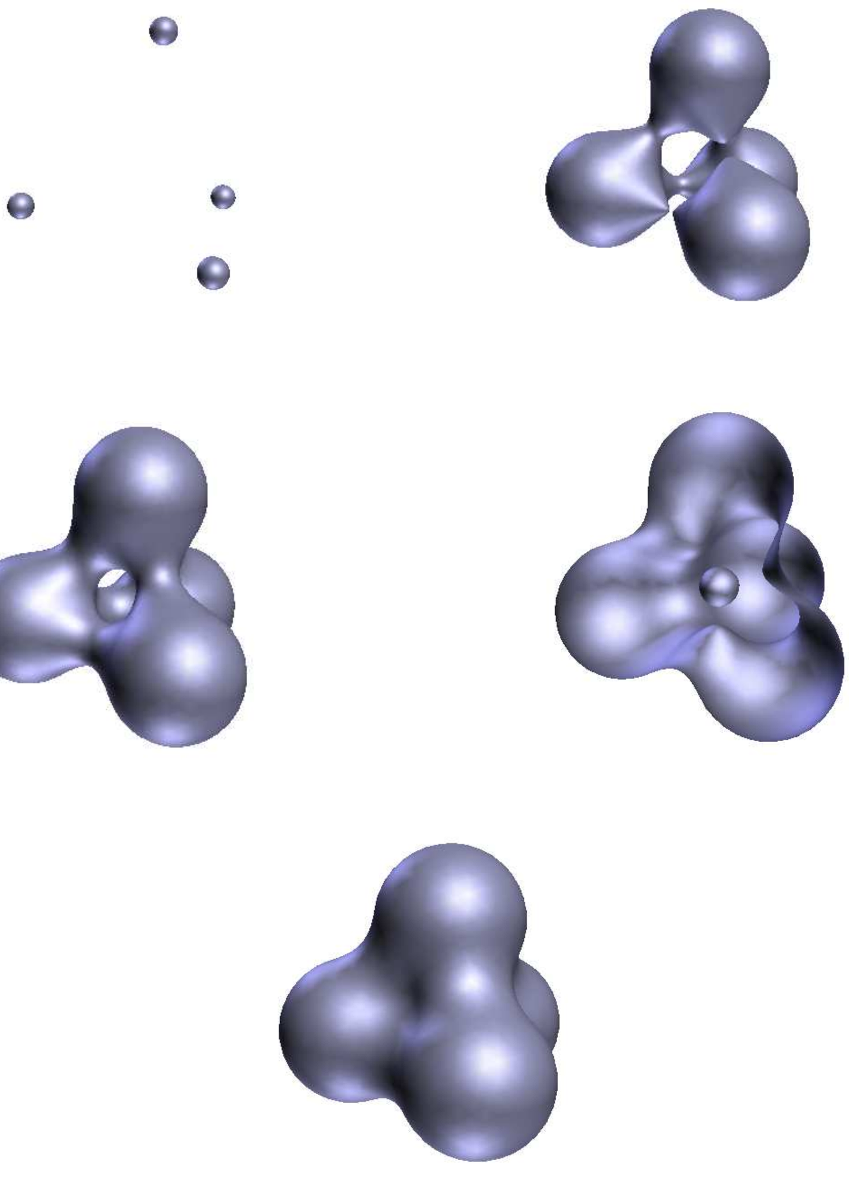


Figure 2.20: The topological changes of the skin surface as we grow the radius simultaneously.

2.3 Triangulations of Skin Surfaces

The goal of this section is to define the triangulation of a skin surface F_B that has the same topology with the surface and approximates the surface accurately as well. First, I introduce the homeomorphism between two sets and define the triangulation of skin surfaces. Then, I describe a special subset of the Delaunay triangulation of the surface samples, namely, the restricted Delaunay triangulation to guide our construction of skin triangulation. Finally, I introduce the conditions that ensure the restricted Delaunay triangulation is homeomorphic to the skin surface.

2.3.1 Homeomorphism

A function g from a set Y to Z is called a *map* if it is *continuous*. The map g is a *homeomorphism* if it is bijective and has a continuous inverse. If a homeomorphism exists from Y to Z then Y is *homeomorphic* to the set Z, denoted as $Y \approx Z$. For example, a square is homeomorphic to a circle, and a cube is homeomorphic to a tetrahedron as well, which are showed in Figure 2.21 (a) and (b) respectively. For another example, a sphere without its north pole is homeomorphic to a plane. The homeomorphism can be realized using the so-called stereographic projection, $g(x) = \frac{x}{1-\|x\|}$, in which x is any point on the sphere except the north pole. See Figure 2.21 (c).

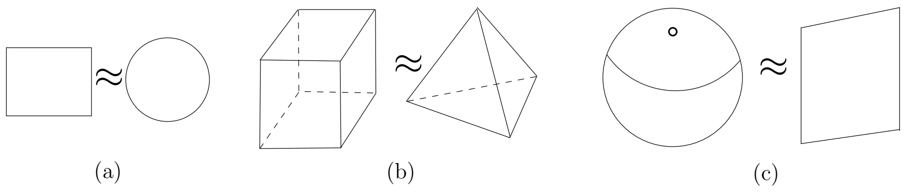


Figure 2.21: Examples of homeomorphic figures.

The *triangulation* of the skin surface F_B is a simplical complex K whose underlying space is homeomorphic to F_B, namely, $|K| \approx F_B$. Figure 2.22 (a) illustrates a homeomorphism between a sphere and an inscribed tetrahedron. By projecting each edge of the tetrahedron from the center onto the sphere, we obtain four spherical patches which are the homeomorphic images of the four faces of a tetrahedron. See Figure 2.22 (b).

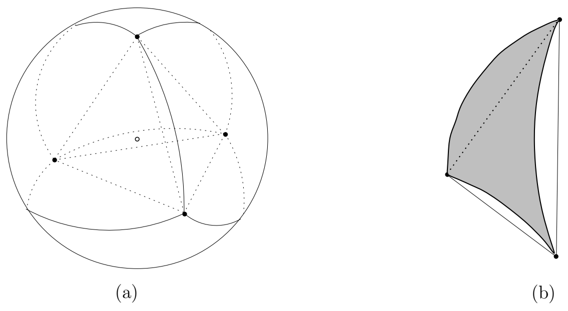


Figure 2.22: The homeomorphism between a sphere and an inscribed tetrahedron and the mapping from a spherical patch to a triangle.

For a skin surface with arbitrary topology, we seek to use a subset of the Delaunay triangulation of the sample points $T \subset F_B$ to construct its triangulation. With the recent results from Edelsbrunner and Shah [50], there is a homeomorphism between the *restricted Delaunay triangulation* and a compact manifold if certain conditions are satisfied. I will introduce the definition of the restricted Delaunay triangulation and the conditions in the next section.

2.3.2 Restricted Delaunay Triangulation

I firstly formalize the definition of the restricted Delaunay triangulation of a skin surface. Then, I describe the closed ball property that ensures the homeomorphism between the restricted Delaunay triangulation and the surface.

Let $T \subseteq F_B$ be a finite subset of points on the skin surface. The *restricted Voronoi polygon* of $a \in T$ is defined as ${\nu_a}' = \nu_a \bigcap F_B$, in which ν_a is the Voronoi cell of a with respect to T in $\mathbb{R}^3$. The nerve of the restricted Voronoi polygons is the set of these polygons with non-empty common intersection. The collection of the convex hull of the nerve element is the *restricted Delaunay triangulation*, namely,

$$D_T{}' = \left\{ \text{conv } (U) \quad \mid \quad U \subseteq T, \quad \bigcap_{a \in U} \nu_a \cap F_B \neq \emptyset \right\}.$$

We assume that there are no four restricted Voronoi polygons with common intersections. It follows D'_T is a simplicial complex consisted of vertices, edges and triangles. See Figure 2.23 for an example.

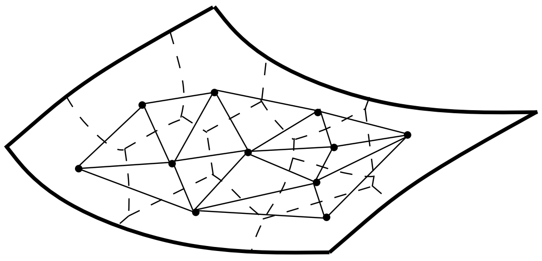


Figure 2.23: The restricted Delaunay triangulation of a partial sampling on a surface. The dashed lines are the restricted Voronoi polygons and the solid lines are the Restricted Delaunay triangulation.

The restricted Delaunay triangulation D'_T is a triangulation of F_B if D'_T has the closed ball property. The closed ball property refers that the intersection of the restricted Voronoi polygons with the surface is a d-dimensional closed topological ball, in which d equals 3 minus the dimension of the Voronoi cell, k. With assumption of general position, we have four cases of Voronoi cells, that is, Voronoi polyhedra, polygons, edges and vertices when $k = 0, 1, 2,$ and 3 respectively. Thus, we can formulate the closed ball property in term of these four cases, that is, the intersection with the skin surface is a closed disk when $k = 0$, empty or a closed interval when $k = 1$, empty or a single point when $k = 2$, empty when $k = 3$. Figure 2.24 illustrates the three possible intersections in three dimensions.

To maintain the closed ball property, the local density of the vertices in the triangulation need to be sufficiently dense. More precisely, we require the set of mesh

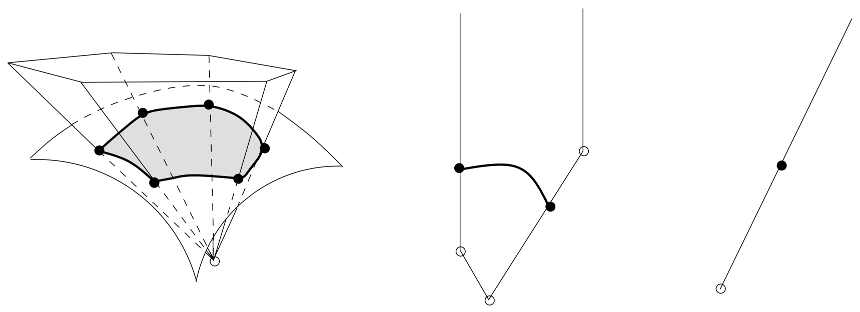


Figure 2.24: The closed ball property in $\mathbb{R}^3$.

vertices is an ε-sampling of skin surfaces with feasible ε value, which will be further discussed in Chapter 4.

2.4 Summary

This chapter introduced the concepts of the simplicial complex, Voronoi diagram and Delaunay triangulation, alpha complex, and defined the skin surface and its triangulation. The properties of the skin surface were reviewed. These background lays the foundations to rest of this thesis. It enables us to develop algorithms for meshing the skin surfaces.

Chapter 3

Adaptive Sweeping Skin Meshing Algorithm

This chapter develops an adaptive sweeping algorithm for meshing the skin surface with guaranteed quality efficiently. The algorithm adapts the advancing front method and sweeps the triangulation from the bottom to the top of a skin surface. The front collision problem during the triangulation sweeping is handled by utilizing the recent result from the computational topology. In particular, a set of *critical points* of a height function on the surface is computed to detect the potential front collisions. The *noisy critical points* are removed using *Morse-Smale complex* to simplify the topological changes of the front. Our implementation results suggest that the algorithm improves the efficiency of the advancing front methods dramatically. Moreover, the triangulation is adaptive to the surface curvature and has a provable lower bound of $21°$ on its minimum angle, which implies a high triangulation quality.

Section 3.1 introduces the Morse function and its critical points on a surface to handle the front collision problem in advancing front methods. Section 3.2 computes the critical points of the height function. The noisy critical points are removed in pair using Morse-smale complex in Section 3.3. Section 3.4 describes the algorithm.

I demonstrate some triangulation results in Section 3.5 and conclude the chapter in Section 3.6.

3.1 Front Collision Handling

First, I review the front collision problem in advancing front meshing. Then, I introduce the Morse function and its critical points, which enable to detect the front collision efficiently when the front is advancing along the direction of the function.

3.1.1 Front Collision Problem

The advancing front methods construct a surface mesh by iteratively attaching triangles to the partial mesh. The mesh boundary, namely, the *front*, is composed of a set of closed piecewise linear curves. In each step, we select a vertex from the front as the *departure vertex*. New mesh vertices are placed around the departure vertex and more triangles are attached to the front using the new vertices. Then the front advances towards the unmeshed region progressively until the whole surface is covered. Figure 3.1 (b) illustrates the updated front after inserting a new mesh vertex and two triangles to the partial mesh in Figure 3.1 (a).

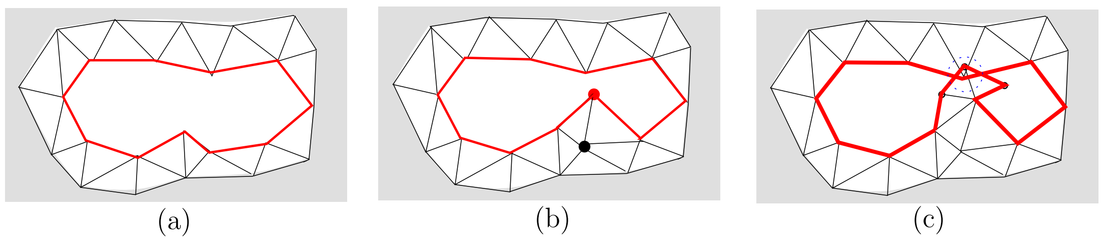


Figure 3.1: Advancing front meshing and front collision problem.

The front collision problem arises when different portions of the front are close to each other. In this situation, the newly added triangles may overlap other existing triangles. See Figure 3.1 (c) for an example. The detection of these collisions is expensive because they may happen anywhere. A number of heuristics were employed to address this problem and these heuristics always lead to unrobust implementation and lower efficiency [60, 66]. For example, Hartman [60] handle this problem by frequently checking the distance of any two vertices on the front, which are costly computations. Even though the collisions can be detected and the front stops advancing at such regions, there will be cracks in the partial triangulations when no more triangles can be added. Fixing these cracks involves heuristics, which always lead to robustness problems of algorithms.

I introduce the Morse theory to handle this problem. Morse theory was developed by Marston Morse in 1930s [79]. Recently, it is applied in the applications such as topological modeling for visualization [56], topological simplification of piecewise linear 2-manifolds and 3-manifolds [43, 44]. Different from these previous works that apply the Morse theory to piecewise linear approximations of smooth manifolds, I will use the *critical points* of a *Morse function* defined on the smooth surfaces to detect the potential front collision in skin meshing algorithm.

3.1.2 Topological Changes of the Front

Morse theory on surfaces describes the topological changes of the partial surface boundary specified by a function, namely, the Morse function [77]. First, I introduce the Morse function and its critical points on surfaces. Then, I describe the relationships between the critical points and the topological changes of the front.

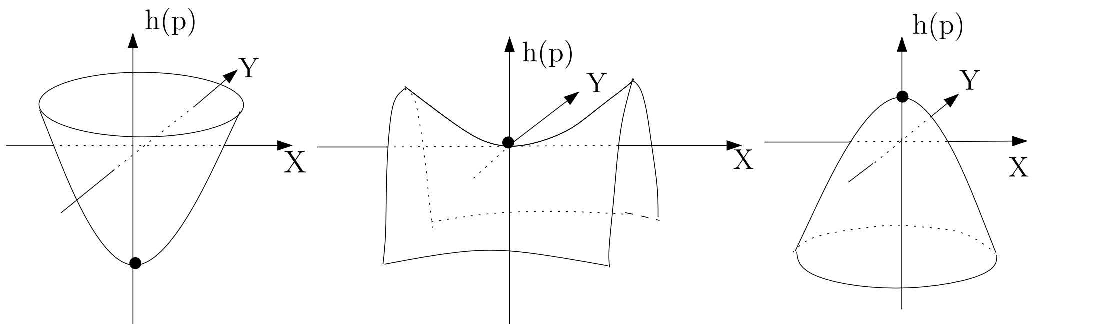
Morse function. We denote $\mathbb{M}$ as a smooth, compact 2-manifold without boundary in $\mathbb{R}^3$ and a function $f : \mathbb{M} \to \mathbb{R}$. We assume a local coordinate system (x_1, x_2) in a neighborhood of a point $p \in \mathbb{M}$. The point p is a *critical point* of function f if all its partial derivatives vanish with respect to the local coordinate system, that is,

$\frac{\partial f}{\partial x_1}|_p = \frac{\partial f}{\partial x_2}|_p = 0$. Otherwise, it is a *regular* point. If p is a critical point, $f(p)$ is the *critical value* of f at p. The critical point p is *non-degenerate* if the *Hessian* of f at p, $H(p)$, that is,

$$H(p) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2}(p) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(p) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(p) & \frac{\partial^2 f}{\partial x_2^2}(p) \end{pmatrix},$$

is non-singular, namely, det $H(p) \neq 0$. The function f is called a *Morse function* if all its critical points are non-degenerate.

For a critical point $p \in \mathbb{M}$, we can choose an appropriate local coordinate system (x_1, x_2) in the neighborhood of p such that the Morse function f is expressed in the form $f(x_1, x_2) = \pm x_1^2 \pm x_2^2 + f(p)$. The number of minuses is called the *index* of f at p. Thus, there are three types of critical points on a smooth 2-manifold in $\mathbb{R}^3$, namely, minima with index 0, saddle points with index 1 and maxima with index 2 [77]. Figure 3.2 illustrates the three types of critical points.



$h(p) = X^2 + Y^2 + c_1$ $\qquad$ $h(p) = X^2 - Y^2 + c_2$ $\qquad$ $h(p) = -X^2 - Y^2 + c_3$

(a) Minimum point $\qquad$ (b) Saddle point $\qquad$ (c) Maximum point

Figure 3.2: Three types of critical points on a 2-manifold.

In this thesis, I employ the height function $h(p) = y_p$ on $\mathbb{M}$ as the Morse function if $p = (x_p, y_p, z_p)$. The critical points of h are the points with horizontal tangent planes.

Topological changes of the front. Let $\mathbb{M}_a = \{x \in \mathbb{M} \mid h(x) \leq a\}$ be the partial surface for some $a \in \mathbb{R}$. Denote $L_a = \{x \in \mathbb{M} \mid h(x) = a\}$ as the *level curve* at a. The curve L_a is the boundary of M_a and it is the intersection of $\mathbb{M}$ and the horizontal plane $\mathbb{T}_a : y = a$.

We illustrate the topological changes of the level curve L_a at the three types of critical points of h with the example in Figure 3.3. Let ϵ be a small positive number. While we sweep a horizontal plane $\mathbb{T}_a$ upwards from $a = -\infty$, the partial surface, $\mathbb{M}_a$, is empty when $a < h(p)$. After $\mathbb{T}_a$ passes the minimum p and when $a = h(p) + \epsilon$, the level curve L_a changes from $L_{h(p)-\epsilon} = \emptyset$ to $L_{h(p)+\epsilon}$, which is a topological circle. When the horizontal plane rises to $\mathbb{T}_{h(r)}$, the two topological circles in $L_{h(r)-\epsilon}$ touch at r. At the level $h(r) + \epsilon$, the curve $L_{h(r)+\epsilon}$ becomes one topological circle. As the height increases to $h(s)$, $L_{h(s)}$ changes to two topological circles with their contact point at s. When the horizontal plane $\mathbb{T}_a$ arrives at the maximum u, one topological circle converges to the point u. Finally, $L_{h(t)+\epsilon}$ becomes empty after the horizontal plane passes the maximum t. To sum up, a component of the level curve L_a is created at the minimum points, two components of L_a are connected or a component of L_a is split to two at the saddle points, and a component of L_a converge to a point at the maximum points.

By this observation, we are inspired to sweep the partial triangulation over the skin surface in a similar manner. The topological changes of the level curve correspond to the topological changes of the front. Therefore, the front will collide around the critical points on the skin surfaces. We set a *protecting sphere* with the center at each critical point to detect the potential collisions of the front. Thus, we only manage the topological changes of the front within each protecting sphere. There are three kinds of topological changes corresponding to the three types of critical points, namely, "creation" at the minima, "bridge" at the saddle points and "seal" at the maxima. Since the number of the critical points are much small comparing with the vertices

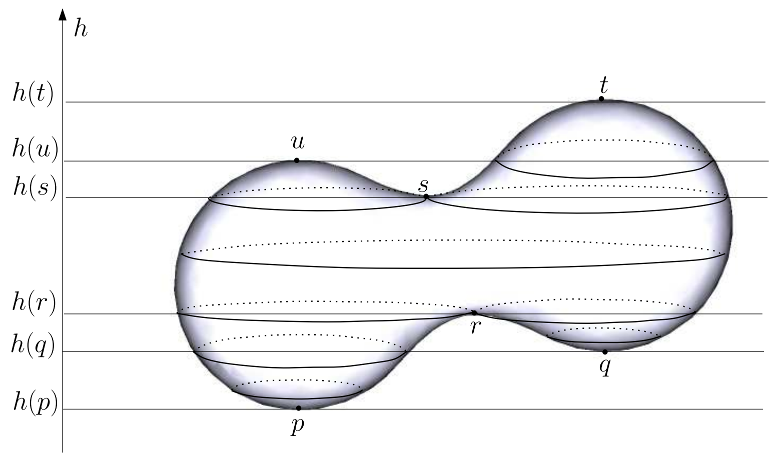


Figure 3.3: Critical points and level curves of height function h on a smooth 2-manifold. The points p, q are minima, r, s are saddle points and t, u are maxima of h. Solid curves are the level curves at different height.

on the front, it would improve the efficiency of front collision handling procedure. In the next section, we compute the critical points on the skin surface analytically.

3.2 Critical Points Computation

The computation of the critical points includes the locations of all the critical points on the skin surface and their types, namely, minimum, saddle or maximum. Since a skin surface can be decomposed to a collection of sphere patches and hyperboloid patches, we can find the critical point for each patch individually. Moreover, the type of the critical point can be determined by the type of the quadratic patch. I first describe the classification of the critical points. Then, I locate the critical points by solving quadratic equations. The correctness of the computation is justified by verifying the Euler characteristic of the skin surface.

Classification of Critical Points. The skin surface clipped within a mixed cell of dimension 0 and 3 is a sphere patch. Therefore, the critical points in such mixed cells can only be a minimum point when its height value is smaller than that of the mixed cell center, and be a maximum point when its height value is larger than that of the center. The skin surface in the mixed cells of dimension 1 and 2 is a hyperboloid patch. In these situations, there are two cases, the skin patch is a one sheeted hyperboloid when the radius of the orthogonal sphere of the Delaunay simplex is positive, or a two sheeted hyperboloid otherwise. The critical points on a one sheeted hyperboloid are all saddle points. And the critical points on a two sheeted hyperboloid are minimum points or maximum points, which can be determined by comparing the height of the critical points and the center of the mixed cell.

Locating Critical Points. The critical points of the height function on the skin surface are the points with horizontal tangent planes. In other words, the surface normal at the critical point is perpendicular to the horizontal plane. We can find such points on each sphere or hyperboloid by solving a quadratic equation. Then, we judge if these points are on the skin surface.

Recall that if we translate and rotate the mixed cell such that its center z_X is the origin, the sphere and hyperboloid can be expressed in a standard form, that is,

$$\begin{aligned} X^2 + Y^2 + Z^2 &= R^2, \\ X^2 + Y^2 - Z^2 &= \pm R^2. \end{aligned}$$

Therefore, the critical points on a sphere are simply the "south pole", $p_s(0, 0, R)$ and "north pole", $p_n(0, 0, -R)$ in this coordinate system. For a hyperboloid, there are four points with horizontal tangent planes. Only two of them can be critical points. Moreover, these two points lie on a hyperbola that is the cross-section of the plane passing through the axis Y and Z. See Figure 3.4 for an example of a two-sheeted hyperboloid case. The cross-section hyperbola can be expressed by $Y^2 - Z^2 = R^2$ in the standard form. The vector $n(n_Z, n_Y)$ is the normalized vector of the height function and the tangent vector at each point p on the hyperbola is $t(1, \frac{Z}{\sqrt{R^2+Z^2}})$. As

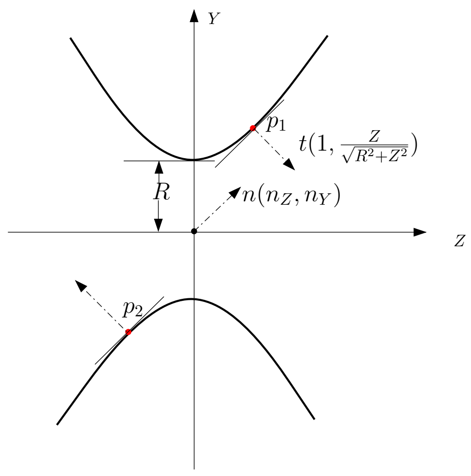


Figure 3.4: Critical points on a two-sheeted hyperboloid.

a result, the coordinates of the two possible critical points p_1 and p_2, whose tangent vectors are perpendicular to the height function direction, are the solutions of the equation $\langle n, t\rangle = 0$, that is, $p_1(\frac{|n_Z|\cdot R}{\sqrt{n_Y^2 - n_Z^2}}, \frac{|n_Y|\cdot R}{\sqrt{n_Y^2 - n_Z^2}})$ and $p_2(-\frac{|n_Z|\cdot R}{\sqrt{n_Y^2 - n_Z^2}}, -\frac{|n_Y|\cdot R}{\sqrt{n_Y^2 - n_Z^2}})$.

Next, we can get the coordination of the two points with horizontal tangent planes on the skin surface by transforming the points p_1 and p_2 to the original coordination system.

Finally, we judge if the points p_1 and p_2 are on the surface. Since each mixed cell is a convex polyhedron, we can always find a point c that is inside the mixed cell. By computing the dot product of the vectors from p_1 and p_2 to c and the normal vector of each face of the mixed cell, we can easily find out whether the points p_1 and p_2 are critical points on the surface or not. If the points p_1 or p_2 are on the face of the mixed cell, they are degenerate critical points. We can simply discard them since they are a pair of critical points that cancel each other. And there are no topological changes of the front at such points.

Euler Characteristic. An easy but effective way to verify the correctness of the critical points computation is to compute the Euler characteristic of the skin surface. For a smooth 2-manifold with g genus, the Euler characteristic is $2-2g$ and it is also the alternating sum of the critical points, that is,

$$\begin{aligned}\chi &= n_{min} - n_{sad} + n_{max}, \\ &= 2 - 2g,\end{aligned}$$

in which n_{min}, n_{sad} and n_{max} are the numbers of minimum critical points, saddle points and maximum critical points respectively [77].

We can compute the Euler characteristic of a skin surface F_B efficiently according to the homotopic equivalence relationship between the skin body and the dual complex of K_B. We have the following Lemma.

Lemma 3.2.1 *Let χ_s be the Euler characteristic of the skin surface F_B and χ_d be the Euler characteristic of dual complex K_B of the union of balls in B, then,*

$$\chi_s = 2\chi_d.$$

PROOF. Denote the Betti numbers of the dual complex K_B as β_{d0}, β_{d1} and β_{d2}. β_{d0} is the number of connected components, β_{d1} is the number of independent tunnels, and β_{d2} is the number of voids. Similarly, the Betti numbers of the corresponding skin surface are β_{s0}, β_{s1} and β_{s2}.

Recall that the skin body is homotopic to the underlying space of the dual complex. Thus, the skin body has the same Betti numbers as that of the dual complex. Since each connected component or void in the skin body corresponds a component in the skin surface, we have,

$$\beta_{s0} = \beta_{d0} + \beta_{d2}.$$

Moreover, each component in the skin surface is a closed surface. Thus, it corresponds a void as well, that is,

$$\beta_{s2} = \beta_{d0} + \beta_{d2}.$$

And each tunnel in the skin body corresponds two tunnels in the skin surface, that is,

$$\beta_{s1} = 2\beta_{d1}.$$

According to the $Euler - Poincare$ theorem [77], the Euler characteristic of the skin surface χ_s is,

$$\begin{aligned}\chi_s &= \beta_{s0} - \beta_{s1} + \beta_{s2}, \\ &= \beta_{d0} + \beta_{d2} - 2\beta_{d1} + \beta_{d0} + \beta_{d2}, \\ &= 2(\beta_{d0} - \beta_{d1} + \beta_{d2}), \\ &= 2\chi_d.\end{aligned}$$

□

The Euler characteristic of dual complex K_B is the alternating sum of the number of vertices, edges and triangles. As a result, we can easily get the Euler characteristic of the skin surface according to the Lemma 3.2.1, which enable us to test the correctness of the set of critical points.

However, two critical points can be close to each other on the portion of the surface with small fluctuations. In this case, the protecting spheres can have non-empty intersection. In order to handle the topological changes of the front, we need to decrease the radii of the protecting spheres until no two protecting spheres intersect. This solution needs to decrease the size of all the triangles in the triangulation as well because the radii of the protecting spheres are proportional to the triangle size. Since the distance between two critical points on a skin surface can be arbitrary small, it would result in huge number of triangles, which would decrease the efficiency of the algorithm and increase the storage.

On the other hand, we can solve this problem in a systematic way. The idea is to remove these critical points that are close to each other in a consistent manner such that the topological changes of the front are captured by the left critical points. Since these critical points close to each other are caused by the spurious features like small fluctuations on the surface, these fluctuations can be approximated by a few fat triangles in the final surface triangulation. In other words, there will be no critical points in such portion of the surface if we define the height function on the surface triangulation. Therefore, we can simplify the height function by removing pairs of critical points such that the height function is equivalent to the height function defined on the skin mesh. Since the critical points that are close to each other represent the spurious features on the surface, I call them *noisy critical points.* In the next section, I will introduce the Morse-Smale complex to remove such critical points.

3.3 Noisy Critical Points Removal

Two noisy critical points can be eliminated by contracting an arc in the Morse-Smale complex [46]. We can remove all the noisy critical points by repeating the arc contraction in the Morse-Smale complex. Edelsbrunner et al. [46] used this idea to construct a hierarchical representation for piecewise linear approximations of the smooth surfaces. In this section, I will first introduce the definition of the Morse-Smale complex. Then, I describe the construction of the Morse-Smale complex on a skin surface. Finally, I introduce the noisy critical points removal by contracting arcs in the Morse-Smale complex.

Morse-Smale complex. The Morse-Smale complex of a Morse function on a smooth 2-manifold decomposes the surface into quadrangular regions whose boundary consisting of exactly four critical points, a minimum, two saddles and a maximum point. The critical points are connected in the Morse-Smale complex via the *integral lines.* An integral line, γ, is a curve on $\mathbb{M}$. For each point $p \in \gamma$, its tangent vector on γ agrees with the gradient of the height function h at p. The *gradient* of the height function h at a point p can be defined as $\nabla h = (\frac{\partial f}{\partial x_1}(p), \frac{\partial f}{\partial x_2}(p))$ in an orthogonal local coordinate system (x_1, x_2). The gradient vanishes at the critical points. For a regular point, the gradient is the tangent vector in the steepest ascending direction.

We can trace out an *integral line* from any non-critical point along the gradient. It starts at a critical points and ascends monotonically. The integral line ends at another critical point. Two integral lines are either disjoint or the same. For a critical point a, the union of all the integral lines ending at a forms the *stable manifold* of a. Specifically, the stable manifold of a minimum point is the minimum point itself, that of a saddle point is open interval, and that of a maximum point is an open disk. The collection of the stable manifolds of all the critical points forms a complex that decomposes M into open cells. Symmetrically, we define the *unstable manifold* of the critical point a as the union of all the integral lines starts at a. We assume the stable and unstable manifold intersect only transversally, that is, a stable and unstable manifold cross at a saddle point when they intersect. This assumption can be easily fulfilled by a small perturbation of the Morse function.

The Morse-Smale complex is the collection of the connected components after intersecting the stable and unstable manifolds for all the critical points on M. It consists of vertices, arcs and regions. Each vertex is a critical point, and each arc is an integral line connecting a saddle point to a minimum or a maximum critical point. The arcs divide the manifold $\mathbb{M}$ into quadrangular regions. Each region is surrounded by two saddle points, a minimum and a maximum critical point [44].

Construction. We construct the Morse-Smale complex of the height function h on the skin surface F_B. Since we have computed all the critical points of h on the surface, the remaining work is to compute the arcs that connecting the critical points. According to the combinatorial structure of the Morse-Smale complex, we can compute the arcs by tracing four integral lines start at each saddle point. First, we determine the tangent vector at each saddle point that indicates the steepest ascent and steepest descent directions. Then, we compute the gradient vector at each regular point on the surface. Finally, we trace the integral curves in an incremental manner.

There are a circle of tangent vectors at each saddle point s and two of them indicates the steepest ascending, denote as u_1, u_2. Because each saddle point lies on an one-sheeted hyperboloid, the two ascending integral lines start from s are the two intervals in a small neighborhood of s, which is the intersection of the skin surface F_B and the plane $\mathbb{H}$ that is parallel to the y-axis and pass through s and the center of the mixed cell contains s. Therefore, the two initial vectors u_1, u_2 of $s \in F_B$ lying on intersection of the plane $\mathbb{H}$ with the tangent plane of $s \in F_B$. Another two tangent vectors, d_1, d_2, that indicate the steepest descending can be obtained easily because they are perpendicular to u_1, u_2 respectively. We call the tangent vectors u_1, u_2, d_1, d_2 the initial vectors of a saddle point.

Then, we compute the gradient of the height function h at the regular points analytically. For each regular point $p \in F_B$, we denote the gradient vector of the height function h at p as g_p and the unit surface normal vector of p as n_p. The height function can be represented by a unit vector $V = (0, 1, 0)$. For each non-critical point $p \in F_B$, the gradient vector, g_p, is the projection of normal vector n_p on the tangent plane of p along the height direction, which is described in the following Lemma.

Lemma 3.3.1 *The gradient vector of height function h at each non-critical point*

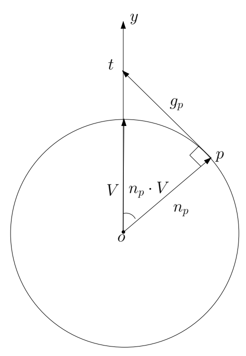


Figure 3.5: Gradient vector of the height function.

$p \in F_B$ is

$$g_p = \frac{V}{n_p \cdot V} - n_p.$$

PROOF. According to the decomposition of the skin surface, a non-critical point p lies on either a sphere or a hyperboloid.

Consider the case that p lies on a sphere. We translate the coordinate such that the center of the sphere is the origin and take the sphere as a unit sphere $\mathbb{S}$. The gradient vector g_p is the tangent vector pt on the circle that is the intersection of the sphere $\mathbb{S}$ with the plane that is parallel to the y-axis and passes through p and o. The point t is the intersection of y-axis with the tangent vector. It is illustrated in Figure 3.5. With the vector subtraction, we get $pt = ot - op$. Since $\mathbb{S}$ is a unit sphere, the vector op is same as the normal vector n_p. Thus, the vector op is perpendicular to the tangent vector pt. And the angle between vector op and ot, $\angle top$, is equal to $\arccos(n_p \cdot V)$. Therefore, the vector $ot = \frac{V}{n_p \cdot V}$. As a result, $g_p = \frac{V}{n_p \cdot V} - n_p$.

If p lies on a hyperboloid, we have an osculating sphere of the skin surface at p. The gradient vector g_p and the surface normal vector n_p at p is same as that of the osculating sphere. The lemma follows the proof in the case p lies on a sphere.

As a result, for any non-critical point $p \in F_B$, the lemma is correct. ◻

Finally, we approximate the integral lines with piecewise linear curves because the integral lines on the skin surface are difficult to compute analytically. At each saddle point, starting from the tangent vectors indicating the steepest directions, we trace four integral lines by iteratively stepping forward following the gradient until we meet a maximum or a minimum critical point. Figure 3.6 illustrate the procedure of tracing an integral line from saddle point s. With point p_1 as the initial position, we step forward a small step size Δs along its gradient vector and get another point q_2. We project point q_2 to the skin surface F_B by computing the intersection of F_B and the line passing through q_2 along the normal vector at p_1. We get the point $p_2 \in F_B$. Repeat this operation until we meet a minimum point or maximum point. The other three integral lines can be computed with the same way. The step size is

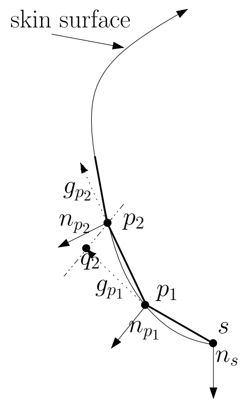


Figure 3.6: Extending an integral line from a saddle point.

important for the accuracy of the computation result. We use the step size adaptive to the maximum curvature at the skin surfaces. We observe that for a point t, the step size use the value $0.2\varrho(t)$ is a feasible value. In the degenerate cases, that is, the integral line meets a saddle point before arriving at a minimum or maximum point, we can do a small perturbation before the integral line crosses the saddle points to ensure the integral lines end at a minimum or maximum point. The accuracy of the approximation is guaranteed by choosing sufficiently small step sizes adaptive to the surface curvature. The Figure 3.7(a) illustrates an example of the Morse-Smale complex constructed on two skin surfaces.

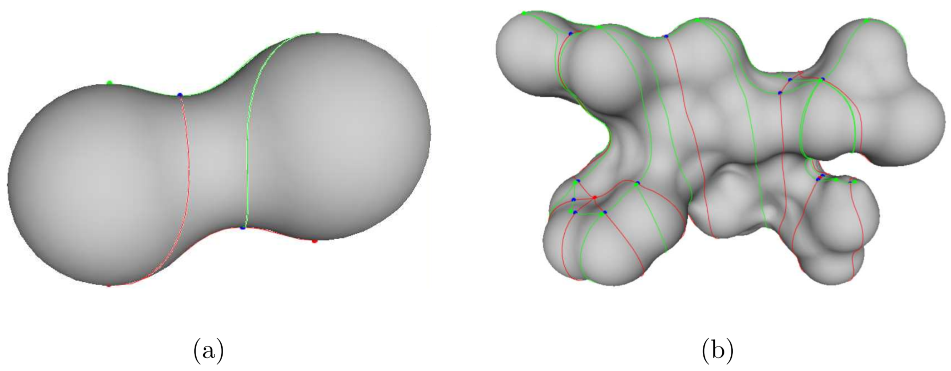


Figure 3.7: The Morse-Smale complex on two skin surfaces. Figure (a) shows the Morse-Smale complex on the surface in Figure 3.3. Figure (b) illustrates the Morse-Smale complex on a molecular skin model of "pdb7tmn". The red points are the minima, the blue points are the saddle points and the green points are the maxima. The red and green curves are the ascending and descending integral lines respectively.

Elimination of the noisy critical points. For the critical points whose protecting spheres intersect each other, we remove them by contracting the short arcs in the Morse-Smale complex. An arc is contracted through deleting its two ending critical points and re-connecting their neighbors. We describe the contraction of an arc with the example illustrated in Figure 3.8.

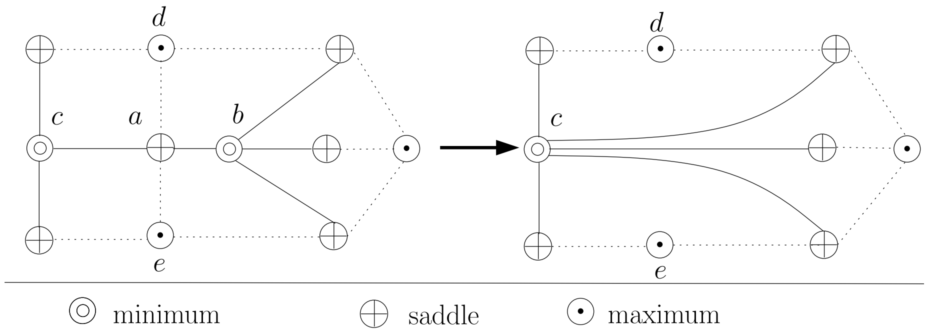


Figure 3.8: The contraction of the arc ab in the Morse-Smale complex.

Let a be a saddle point and b be a minimum critical point. Suppose that the arc ab is a short ascending integral line from b to a. We can view the contraction of the arc ab as merging critical points a, b into another minimum critical point c that connected to a. The four integral lines connected to a are removed and all the integral lines start from b are extended to c. After the contraction of ab, each region is still a quadrangle with minimum, saddle, maximum and saddle in order. The short arcs between the saddle points and maxima can be contracted in a similar way.

Contraction of the short arcs between a saddle point and a minimum or a maximum critical point is the only operation required in eliminating the noisy critical points. With a sequence of contraction operations, it is sufficient to get a simplified Morse-Smale complex in which no two protecting spheres intersect each other [44] .

We explore each saddle point and check if its protecting sphere intersect with its neighbors in the Morse-Smale complex. The arcs between pairs of a saddle and a minimum or a maximum critical point are contracted if their protecting sphere intersects each other. As a result, we get a simplified Morse-Smale complex in which no any two protecting spheres of the critical points intersect each other.

3.4 Algorithm

In this section, I give a complete description of our meshing algorithm. The input of the algorithm is a set of weighted points, B, and the output is a triangulation, K, which approximates the skin surface F_B. I first give an overview of the algorithm. Then, a top-down description of the algorithm and essential operations in the algorithm are introduced. Finally, the curvature adaptive schema is introduced followed by the description of the mesh refinement operations.

3.4.1 Overview

The algorithm adapts the advancing front meshing method and utilizes the critical points of the height function to detect front collisions. In particular, the initial front

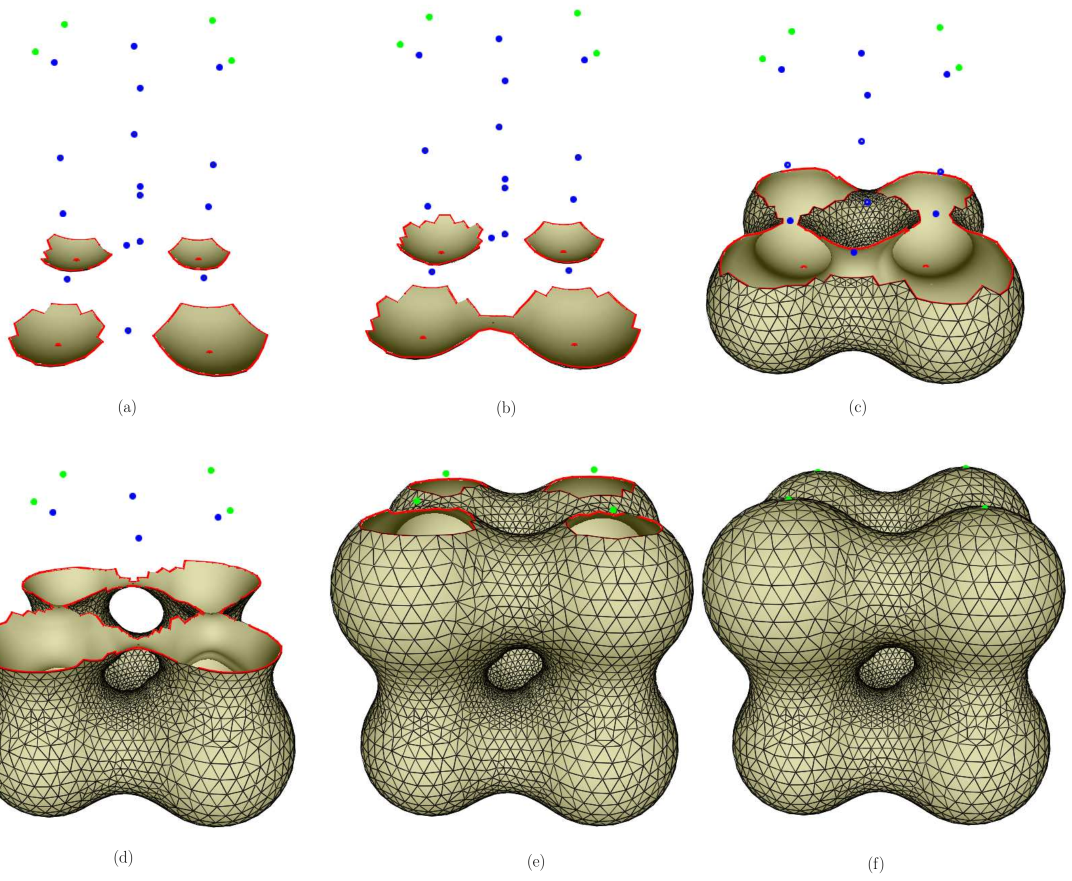


Figure 3.9: Six snap-shots of the growing mesh from the bottom to the top of the skin surface defined by 8 weighted points at the corners of a cube. The red points are the minima, the blue points are the saddle points and the green points are the maxima. The red boundary of the partial mesh is the front.

is created from the minimum critical points. When the front is advancing, the new triangles are always attached to the lowest vertex on the front, which is called *departure vertex*. Thus, the partial triangulation sweeps from the bottom to the top of the skin surface until the whole surface is covered. Figure 3.9 illustrates the six snap-shots of the growing mesh of the skin surface defined by 8 weighted points at the corners of a cube. Moreover, in each step of adding new triangles, their sizes are adaptive to the surface curvature and the triangulation is guaranteed to be homeomorphic to the surface. Next, we give a top-down description of the algorithm and introduce the essential operations in the algorithm.

3.4.2 The Adaptive Sweeping Algorithm

The algorithm is divided into two stages, namely, initialization stage and the stage of sweeping triangulation.

Initialization. In this stage, we construct two combinatorial structures, namely, the mixed complex M_B of the set of weighted point B and the Morse-Smale complex of the height function h on the skin surface F_B. These two complexes guide the mesh generation in the next stage. We first compute the mixed complex, M_B, of B by constructing the weighted Delaunay triangulation of B. For each polyhedron μ in M_B, it is the Minkowski sum of a simplex in the weighted Delaunay triangulation of B and its dual Voronoi cell, scaled by 1/2. Then, we locate all the critical points on each quadratic patch within μ. Finally, we compute the Morse-Smale complex. The Algorithm 3.1 is the detail of the initialization stage.

Algorithm 3.1 Initialization()

```
1: Compute the mixed complex, $M_B$, of $B$;
2: Locate all the critical points on $F_B$:
       a. $C^-$: = the set of minima;
       b. $C^+$: = the set of maxima;
       c. $C^0$: = the set of saddle points.
3: Construct the Morse-Smale complex;
4: Contract short arcs in the Morse-Smale complex;
```

We set a protecting sphere for each the critical point according to their curvature, which will be described in Section 3.4.3. If two protecting spheres intersect, we eliminate the critical points by contracting the short arcs in the Morse-Smale complex.

Sweeping Triangulation. Next, we add triangles to the mesh incrementally. The partial triangulations sweep over the skin surface along the height direction. The sweeping triangulation stage is divided into three steps, namely, *creating the initial* *front, creeping triangles* and *sealing holes*, and they are implemented according to the topological changes of the front at the three types of critical points, namely, minima, saddle point and maxima respectively. The following Algorithm is the detail of the stage of sweeping triangulation.

Algorithm 3.2 SweepingTriangulation()

```
1: CreateInitialFront();
2: CreepTriangles();
3: SealHoles();
```

First, we create the initial front at the minima with the "creation" operations. Then we advance the front along the height direction by attaching triangles to the lowest vertex on the front iteratively. During this process, the different portions of the front connect around the saddle points within its protecting sphere, and we call this the "bridge" operation. When the front is in the protecting spheres of the maxima and no more triangles can be attached, we close the holes with the "seal" operation and the algorithm terminates.

Initial front. We first construct the initial front by applying "creation" operation at each minimum critical point. The "creation" operation creates a "bowl" of 6 neighboring triangles that share one common vertex. See Figure 3.10.

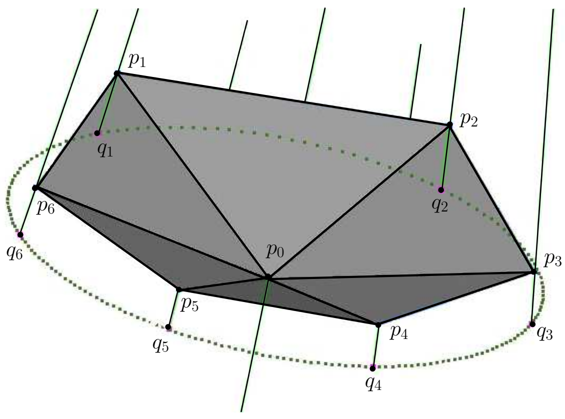


Figure 3.10: Create a bowl at the minimum point p_0.

Let p_0 be the minimum critical point. We first draw a tangent disk with p_0 as the center. Then we locate six points $q_i, i = 1..6$ on the boundary of the tangent disk. See the dashed circle in Figure 3.10. These six points divide the tangent circle into six arcs uniformly, that is, $\angle q_i p_0 q_{i+1} = 60°$. We add the new vertices p_i by projecting q_i to the skin surface and form the six triangles. The boundary of each bowl is a cycle of 6 edges, which is a *front polygon.* After applying *creation*() operation to all minima, we get the initial front, which consists of a set of front polygons. We push all the edges in the front polygons to a priority queue, Q, which enables us to get the departure vertex with the minimum height in the next step.

Creeping triangles. In this step, we add more triangles iteratively from the initial front. There are two operations in this process: the *creep*() operation to attach triangles to the front from a departure vertex, and the *bridge*() operation to connect two portions of the front around the saddle points. The step of creeping triangles is implemented with *CreepTriangles*(), which is illustrated in the following pseudo code.

Algorithm 3.3 CreepTriangles()

```
1: while $Q \neq \varnothing$ do
2:    $t$ = ExtractMin($Q$);
3:    if $t$ falls in the protecting sphere of a saddle point $c$ then
4:       if $c$ is ready to be bridged then
5:          bridge();
6:       else
7:          Mark the triangle $t$ with red;
8:       end if
9:    else if $t$ falls in the protecting sphere of a maximum point $c$ then
10:      Mark the triangle $t$ with green;
11:   else
12:      creep();
13:   end if
14:   Refine the newly added triangles;
15: end while
```

In each iteration, we get the departure vertex from Q to attach new triangles. A fan of triangles around the departure vertex are created by the *creep*() operation.

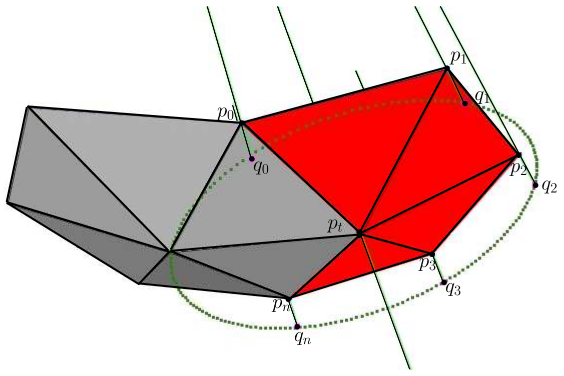


Figure 3.11: Creep triangles from a departure vertex p_t.

See Figure 3.11 as an example. Let p_t be the departure vertex. We first draw the tangent disk with p_t as the center. The dashed circle in Figure 3.11 is the boundary of the tangent disk. Let p_0, p_n be the neighbors of p_t on the front and the points q_0, q_n are the projections of p_0, p_n on the tangent plane of p_t. We add new vertices p_i and form new triangles $p_{i-1}p_tp_i$ from $i = 1$ to $i = n$, in which $n = \lceil \frac{\angle q_0p_tq_n}{60°} \rceil$. To get p_i, we first locate q_i that satisfies $\angle q_ip_tq_{i+1} \simeq 60°$. Then, we project each q_i to a point p_i on the skin surface. After each *creep*() operation, we push the new front edges to the priority queue Q. While the priority queue Q is not empty, we repeat the *creep*() operation to advance the front towards the untriangulated part of the surface.

During the front advancing by the *creep*() operation, the small angle between two front edges may cause overlap between the newly added triangles and the existing triangles. See the Figure 3.12 (a).

In Figure 3.12, the angle $\angle gfd$ is a small angle and the vertex d is the departure vertex. The dashed circle is the boundary of the tangent disk in the current *creep*() operation. In the Figure 3.12 (a), two newly added mesh vertices are p_0 and p_1. Because $\angle gfd$ is small, the vertex p_1 lies in the existing triangle efg. As a result, the new triangles fp_1d and p_1p_0d overlap the existing triangle efg. We can solve this problem by fixing the angle $\angle gfd$ before we apply *creep*() to vertex d. As illustrated in the Figure 3.12 (b), the newly added triangles from d do not overlap the existing

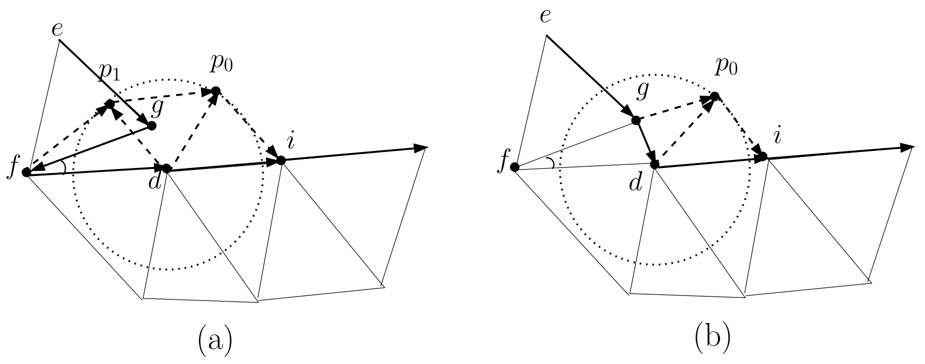


Figure 3.12: Wing a small angle to avoid the overlapping triangles in *creep()* operation. Figure (a) and (b) illustrate the result of *creep()* operation before and after fixing the small angle $\angle gfd$.

ones after adding a triangle gdf.

We call fixing small angles the *wing*() operation and apply it at the end of each *creep*() operation to fix all the small angles. It ensures that there are no angles smaller than 90° between any two adjacent front edges during front advancing. However, the newly added triangles in the *creep*() operation may not have good qualities. We will introduce the refinement operations in Section 3.4.4.

Bridging the front. As the partial mesh grows, the fronts will get closer to each other around the saddle points or the maxima. The protecting spheres for the critical points will detect such events. Once the front falls into the protecting sphere of a saddle point, we stop the front advancing at this portion because it would lead to front collision. If the front reaches the protecting sphere of a saddle point on two different sides, we use a *bridge*() operation to connect the two sides of the front. Figure 3.13 illustrates the result a bridge operation.

In Figure 3.13, the vertex p_s is a saddle point and the thin dotted line represents the protecting sphere of p_s. The front edges $p_l p_r$ and $p_u p_v$ fall in the protecting sphere on two different sides of p_s. We add new triangles to bridge the front together at the edges $p_l p_r$ and $p_u p_v$. Then, we update the priority queue Q after a *bridge*() operation

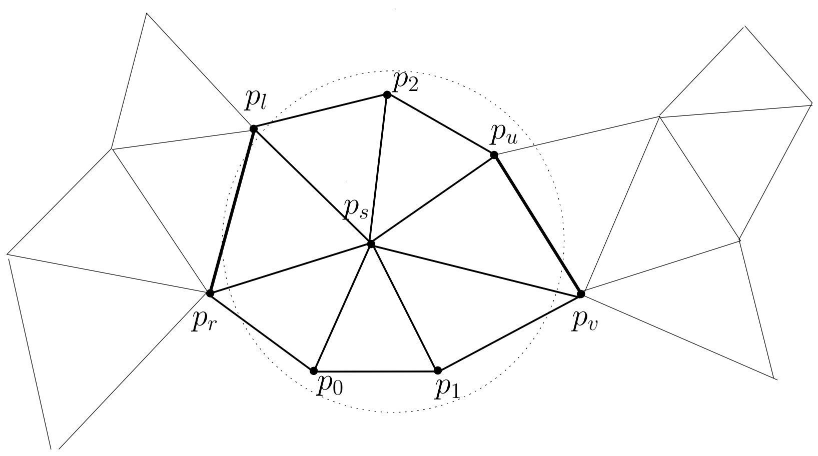


Figure 3.13: Bridge the front at a saddle point p_s.

and resume the loop of *creep*() operation to advance the front. See the snapshots (b), (c) and (d)in Figure 3.9 for an example.

Sealing holes. Once the front falls into the protecting sphere of a maximum critical point, we simply stop the front advancing at that portion. When the priority queue Q is empty, the surface, F_B, is covered by an almost finished partial triangulation with holes only around maxima. The boundary of each hole is a topological circle consisted of a cycle of edges in the protecting sphere of a maximum. We seal the hole by repeatedly adding triangles whose two edges are neighbors in the circle. The snapshots (e) in Figure 3.9 illustrates the holes around each maximum before the sealing operation.

Finally, we get the triangulation K of the skin surface F_B after refining the newly added triangles in these *seal()* operations.

3.4.3 Curvature Adaptation

Adaptation of the surface mesh to the curvature is critical to decrease the number of the triangles in the mesh because we can use large triangles to approximate the region with smaller curvature. In our skin meshing algorithm, we adapt the surface mesh to the maximum curvature of the skin surface. In particular, we control the edge lengthes and circumradii of triangles in the mesh to be proportional to the radius of the maximum curvature, namely, the local length scale at their vertices. Since the edge length and triangle size depend on the radius of the tangent disks we used in the *creation*() and *creep*() operations, we control the mesh size by adapting the radius of the tangent disks to the local length scale at the departure vertex. Also, we use the same radius for the protecting spheres at the critical points. The reason is that each step of the front advancing must not cross to the other side of the spheres.

I first introduce two conditions that bound the edge length and triangle size in skin triangulation proposed by Cheng et al. [23]. Then, I deduce the lower bound and upper bound for the edge length. Finally, I give the curvature adaptation formula of the radius of the tangent disk.

Conditions. Cheng et al. [23] used two conditions to bound the edge lengths and the circumradii of the triangles in the surface mesh such that the surface mesh is homeomorphic to the skin surface and have a lower bound on the minimum angle of the triangles. The condition [L] restricts the edge length not too short, and the condition [U] bounds the circumradius of a triangle not large in terms of the local length scale at their vertices, that is,

[L] $R_{ab} > \frac{C}{Q}\varrho_{ab}$, for every edge ab,

[U] $R_{abc} < CQ\varrho_{abc}$ for every triangle abc.

In Condition [L], notation R_{ab} is the *size* of an edge ab which is equal to $\|ab\|/2$. Similarly, the size of a triangle abc is defined as R_{abc} that is the circumradius of the triangle abc. The local length scale of the edge ab is defined as $\varrho_{ab} = \max\{\varrho(a), \varrho(b)\}$ and that of the triangle abc is defined as $\varrho_{abc} = \min\{\varrho(a), \varrho(b), \varrho(c)\}$. C and Q are judiciously chosen positive constants. C controls how closely the mesh approximates the surface and Q controls the quality of the mesh. Cheng et al. [23] also proved that the restricted Delaunay triangulation has the closed ball property if these two Conditions are satisfied with C = 0.08 and Q = 1.65. This implies it is homeomorphic to the skin surface F_B [50]. At the same time, the minimum angle in the surface mesh is larger than $\arcsin \frac{1}{Q^2}$, which is around 21° for Q = 1.65.

In this algorithm, I will control the triangle sizes to maintain these two conditions while constructing the skin triangulation incrementally. For these triangles that violate the conditions [L] and [U], I will apply local refinement operations to force the surface mesh satisfying the conditions, which will be introduced in the next section. As a result, we can achieve homeomorphic skin triangulation with a lower bound on its minimum angle as well. Next, I deduce the lower bound and upper bound of the edge length according to conditions [L] and [U].

Edge length constraints Let the departure vertex be a and a new mesh vertex be b. Denote the radius of the tangent disk as r_a centered at a. Our consideration is to ensure the edge length of ab is not too long nor too short for both vertices a and b according to the two Conditions [L] and [U]. We first derive the lower bound and upper bound of the size of edge ab, R_{ab}, relative to the local length scale $\varrho(a)$.
Lower bound. Starting from a vertex a, it is trivial to see that the length of an edge ab has a lower bound of $2\frac{C}{Q}\varrho(a)$. Although the edge is not too short for the vertex a, it may be too short for the other end b. The reason is that the local length scale of b, $\varrho(b)$, may be greater than $\varrho(a)$, thus, ab may be too short for b. However, we know the bound for $\varrho(b)$ because the change of local length scales of two vertices on $\mathcal{F}$ is less their distance, i.e.

$$\|a - b\| \;\; \geq \;\; |\varrho(a) - \varrho(b)|.$$

It follows that $\varrho(b) \leq \varrho(a) + 2R_{ab}$. In order to ensure the length of ab is not too short for b, the Condition [L], $R_{ab} > \frac{C}{Q}\varrho(b)$, must be satisfied for the vertex b also. Since the worst case is $\varrho(b) = \varrho(a) + 2R_{ab}$, the edge ab is not too short for both endpoints if

$$R_{ab} \quad > \quad \frac{C}{Q-2C}\varrho(a).$$

The inequality tells that we need a slight longer edge in order to make sure the edge is long enough for both vertices a and b.

Upper bound. Similarly, the edge size is also bounded by the Condition [U] since the size of any edge of a triangle cannot exceed its circumradius. It is obvious that the Condition $R_{ab} < CQ\varrho(a)$ must be satisfied in order to ensure the edge is not too long for vertex a. However, the edge may be too long for vertex b because $\varrho(b)$ may be smaller than $\varrho(a)$. Following the curvature variation limitation, the lower bound for $\varrho(b)$ is $\varrho(a) - 2R_{ab}$ and we want to keep the Condition $R_{ab} < CQ\varrho(b)$ also. Thus, we have the new constraint for an edge ab that

$$R_{ab} \quad < \quad \frac{CQ}{1+2CQ}\varrho(a).$$

With the edge length constraints, we can determine the radius of the tangent disk.

Radius of the tangent disk. To ensure the edge is not too short, we can use the bound of $r_a > \frac{2C}{Q-2C}\varrho(a)$ since $R_{ab} > r_a/2$. For the upper bound of r_a, we know that R_{ab} increases when b gets further from the tangent disk. However, there is a *sandwiching ball* with radius $\varrho(a)$ for every vertex a. The sandwiching ball is tangent to the skin surface at the point a and the skin surface does not penetrate into this ball. Figure 3.14 shows the cross section of the skin surface and the sandwiching ball at a.

Then, the longest edge will be created if b is on the surface of the sandwiching

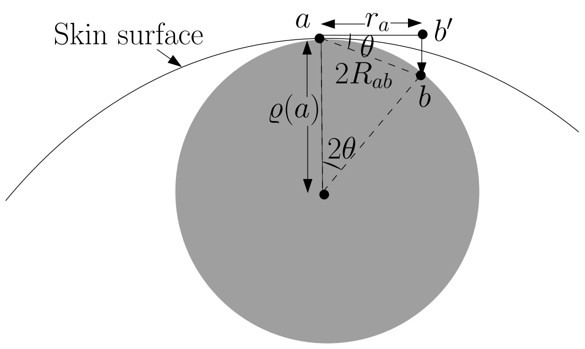


Figure 3.14: Cross section of the scenario of projecting the point from the tangent disk with radius r_a.

ball of a. In Figure 3.14, we compute the angle $\theta = \arcsin(R_{ab}/\varrho(a))$ and thus

$$\begin{aligned} r_a &= 2R_{ab}\cos\theta \\ &= 2R_{ab}\sqrt{1-(\frac{R_{ab}}{\varrho(a)})^2} \end{aligned}$$

Substituting the upper bound of $R_{ab} = \frac{CQ\varrho(a)}{1+2CQ}$ into the above equation, the upper bound of the tangent disk radius is

$$r_a = \Big(\frac{CQ}{(1+2CQ)^2}\sqrt{(1+CQ)(1+3CQ)}\Big)\cdot\varrho(a).$$

For $C = 0.08$ and $Q = 1.65$, the numerical value of $r_a/\varrho(a)$ is around 0.208, which is better than the lower bound $\frac{2C}{Q-2C} = 0.107$.

3.4.4 Local Refinement

The curvature dependent radius of the tangent disk r_a ensures that the edge length is not too long nor too short in most cases. However, the partial triangulations may not maintain the conditions [L] and [U] during sweeping. It is mainly due to three reasons. Firstly, the local region of newly added triangles in each step may not keep the restricted Delaunay property, we use *edge flipping* to maintain the restricted

Delaunay triangulation. Secondly, the short edges added in the *wing()* operation may violate the Condition [L], we use the *edge contraction* to remove these short edges. Finally, edge contraction may lead to violation of the Condition [U]. Thus, we need the *vertex insertion* operation to destroy the big triangles.

We refer to the implementation of the refinement operations introduced in the dynamic skin triangulation algorithm [23]. After each step of adding new triangles, these refinement operations are applied. As a result, we maintain the closed ball property for the sweeping mesh. Cheng et al. proved that these refinement operations terminate, which implies the termination of our algorithm. Experiments with our implementation show that the refinement operations only influence a very small number of triangles around the new attached ones in each *creep*() operation. It means the refinement operations are efficient. At the same time, these refinement operations guarantee the minimal angle in the triangulation is larger than 21°, which implies a good quality mesh.

Since the implementation in [23] assumes the triangulation has no boundary, we extend it in three aspects. Firstly, we restrict that a front edge is not "flippable". Secondly, in the vertex insertion operation, if the vertex x we need insert lies outside of the triangulation, we split the front edge instead of inserting vertex x. Finally, after each refinement operations, we need update the priority queue q according to the changes of the front. Edge flipping operation replace an edge ab by the diagonal of the quadrangle $abcd$. An edge ab is contracted through deleting the vertex b together all the edges and triangles contains b. Insert an vertex on the surface at equal distance from a, b, and c in the triangle abc break a big triangle to three smaller ones.

In each step of adding new triangles, we check if the newly added triangles satisfy the condition [L] and [U]. We push the triangles that break the conditions to three stacks, that is, flip stack, contraction stack and insertion stack. We pop the flip stack first, then the contraction stack and finally the insertion stack. The refinement terminate when all the stacks are empty. Note that there is no infinite loop in the refinement operations [23].

3.5 Results

I demonstrate the skin meshes constructed by our algorithm in this section. Figure 3.7(b) shows the surface of a simple molecule named "pdb7", which is a binding inhibitor of protein molecule. Figure 3.15 (a) shows the molecular skin model of a DNA molecule and the zoomed in view of the mesh details in the rectangle is illustrated in Figure 3.15 (b). Figure 5.5(a) shows the molecular skin surface model of "Gramicidin A" molecule and Figure 3.16(b) illustrates its partial mesh details included in the box of Figure 3.16(a).

Table 3.1 lists the statistical results of the examples shown in this chapter, along with a comparison of computation time with the dynamic skin triangulation algorithm [23]. A Pentium 4 processor running at 2.54GHz is used in the test. As shown in the third column of Table 3.1, we achieve a better bound of the minimum angle of the mesh in practice. For example, the minimum angle in the triangulation of "pdb7" molecule is 27.02°, which is much better than the theoretical result 21°. The high quality we achieved may be explained by three factors. First, the smoothness of the molecular skin model is crucial to generating quality meshes. It would be problematic to guarantee a lower bound on the minimum angle of the triangulation if the surface is not smooth. Second, our strategy of the initial vertices placement in the creeping process creates well-shaped triangles. Third, the mesh refinement operations kill the bad shape triangles.

The results also demonstrates that our algorithm achieved high efficiency. For example, the computing time of the molecule Gramicidia A decreased from more than 1 hour to 3 minutes. The dramatic improvement of the efficiency over the dynamic

molecular name	no. of triangles in the mesh	minimum angle in the mesh	computing time: our approach	computing time: Dynamic skin
$pdb7tmn(Figure3.7(b))$	24,336	27.02°	00:00:05	00:10:00
$A-DNA(Figure3.15)$	114,316	24.12°	00:00:51	00:35:12
$GramicidinA(Figure3.16)$	305,186	24.37°	00:03:22	01:35:23

Table 3.1: Performance of the adaptive sweeping triangulation algorithm.

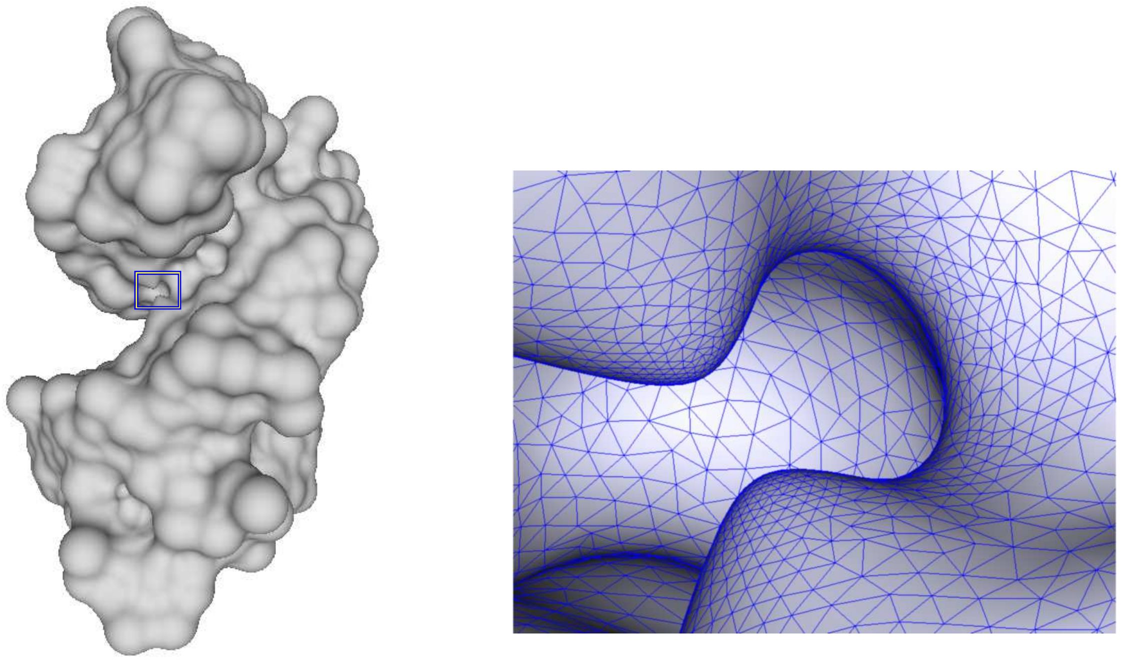
skin triangulation algorithm can be explained by using the advancing front method to avoid the costly computation of 3D Delaunay triangulation. Conventional methods to handle this problem are always time consuming because the potential collisions are detected by frequently checking the distance of any two vertices on the front [60]. Even though the collisions can be detected and the front stops advancing at such regions, there will be cracks in the partial triangulations when no more triangles can be added. Fixing these cracks involves heuristics, which always lead to robustness problems of algorithms.

3.6 Summary

This chapter presents an algorithm for constructing high quality mesh of the skin surface with guarantees on the topology and a lower bound of the minimum angle as 21°. The surface triangulation can support not only the protein visualization but also numerical simulations of the protein interactions. The algorithm achieves high efficiency by employing the Morse theory to handle the front collision problem in advancing front method.

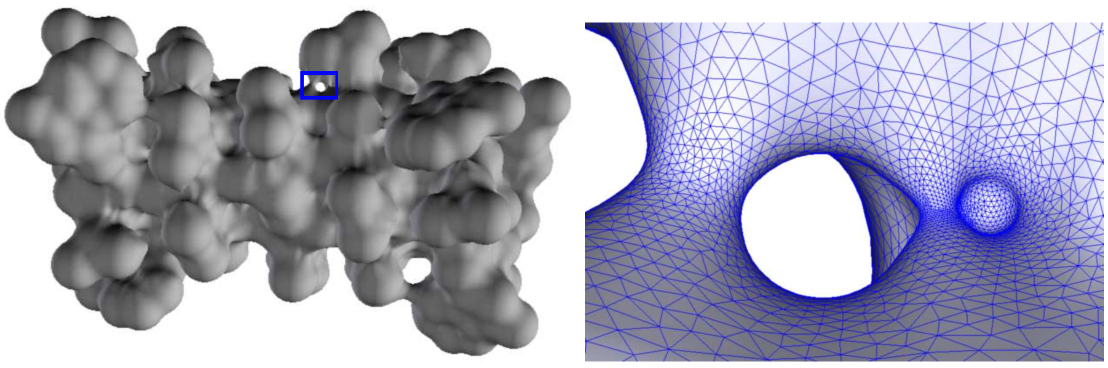
The choice of the height function in this chapter might be questionable as the resulting critical points and Morse-Samle complex are obviously dependent on the choice of the height direction. However, our algorithm can generate correct mesh with different directions of the height function. Another issue is the number of the critical points. Since a large number of critical points may affect the efficiency of our algorithm, we tackle this by using an efficient point location data structure, namely, the *kd-tree*. On the other hand, it is possible to use other Morse functions instead of the height function and we change the priority of the front advancing accordingly.

It should be noted that our application of Morse theory on skin triangulation differs from Stander et al. [92]. On one hand, we use a height function on the surface, a smooth 2-manifold, as the Morse function. Stander et al. use the function that defines the implicit surface as the Morse function, which could be considered as a height function on a 3-manifold. On the other hand, Stander et al. guarantee the homeomorphism between the mesh and the implicit surface by tracking the critical points of the implicit function. However, our homeomorphic triangulation is guaranteed by locally controlling the triangles size. Besides, a related concept to the Morse-Smale complex in this chapter is the Reeb graph used in the surface reconstruction [56]. The Reeb graph is a compressed representation of the components of implicit surfaces. However, the Morse-Smale complex expresses the gradient flow on surfaces. We use the Morse-Smale complex of a height function to simplify the topological changes of the front.



(a) (b)

Figure 3.15: The molecular skin models of A-DNA molecule. The left figure shows molecular skin model and the right shows the zoomed mesh details in the box of the center right figure..



(a) (b)

Figure 3.16: The molecular skin model of Gramicidin A. Figure (a) shows its surface model and (b) illustrates the zoomed mesh details of the portion in the box of (a).

Chapter 4

Skin Meshing Using Restricted Union of Balls

A new skin meshing algorithm that integrates the advancing front methods into Delaunay triangulation is developed in this chapter. The algorithm incrementally samples points on the surface and constructs the Delaunay triangulation simultaneously. By associating each sample point a ball centered on the surface, the intersection of the union of these balls with the skin surface, namely, *the restricted union of balls*, facilitates the point sampling process and guarantees that an even ε-sampling of the skin surface is generated when the algorithm terminates. A subset of the Delaunay triangulation, namely, the restricted Delaunay triangulation, forms a quality mesh of the skin surface. The algorithm not only offers guarantees on both the mesh quality and the homeomorphism between the triangulation and the skin surface but also performs excellently in practice.

Section 4.1 introduces the idea of integrating the advancing front methods into the Delaunay triangulation for the skin meshing. Section 4.2 describes the sampling theory of the skin surface and the concept of the restricted union of balls. Section 4.3 introduces the computation of the connected components for a skin surface. Section

4.4 describes the meshing algorithm using restricted union of balls. Some triangulation results are illustrated in Section 4.5. I conclude the chapter in Section 4.6.

4.1 The New Idea: Advancing Front Meets Delaunay Triangulation

Advancing front methods generate a surface mesh by adding triangles to the front iteratively until the whole surface is triangulated. The advantages of the advancing front methods include high efficiency and precise control over the mesh vertices placement [60, 66, 89, 64]. However, there are two challenges resided in the front advancing methods.

Firstly, it is challenging to handle the front collision problem. Although the Morse function and its critical points enable the algorithm to detect the front collision efficiently, the noisy critical points arising on the bumpy region of the surface complicate this solution. We can apply the Morse-Smale complex to simplify these noisy critical points but it results in robustness problem of the implementation. The reason is that the arcs in the Morse-Smale complex are solutions of partial differential equations (PDEs) with order 2 and the current methods for solving PDEs cannot give analytical solutions for such equations. Thus, we can only approximate the arcs in the Morse-Smale complex by numerical methods. The accumulation of numerical errors in the approximation leads to inconsistent critical points after eliminating noisy critical points inaccurately, which may fail the meshing algorithm described in Chapter 3. For example, the algorithm in Chapter 3 fails to triangulate the molecular skin surface illustrated in Figure 4.1 because the bumpy appearance on the surface resulted in a large number of noisy critical points and a complicated Morse-Smale complex.

Secondly, advancing front methods do not provide any guarantees on mesh quality. As a result, it is necessary to introduce mesh refinement operations to improve

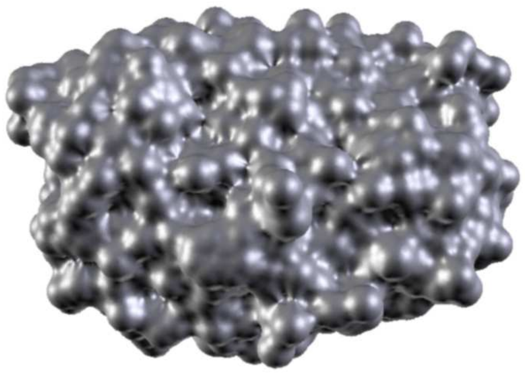


Figure 4.1: The molecular skin model of HIV-2 protease.

the mesh quality as a post processing. The refinement procedure could be costly operations that decrease the efficiency of the algorithm.

On the other hand, Delaunay mesh generators provide provable guaranteed mesh quality and have robust and efficient implementations. Integration of the Delaunay triangulation into advancing front methods suggests a new meshing algorithm that captures the advantages of both front advancing and Delauany-based meshing methods. More precisely, we can use the advancing front methods to generate a good sampling of the surface rather than the mesh itself. At the same time, the Delaunay triangulation of the sample points is computed and a subset of the Delaunay triangulation is used to approximate the surface.

In order to generate high quality surface meshing, the sampling density usually adapts to the surface curvature and guarantees a lower bound on the edge length in the mesh. Next, I will introduce the ε-sampling of the skin surface and the restricted union of balls for the density control in the meshing algorithm. I will also describe the properties of the restricted Delaunay triangulation of the ε-sampling for the purpose of surface mesh extraction.

4.2 Sampling Theory of Skin Surfaces

I will use a special subset of the Delaunay triangulation of the sample points, namely, the restricted Delaunay triangulation as the final surface mesh. In order to guarantee the restricted Deluany triangulation to be homeomorphic to the surface, the closed ball property needs to be satisfied. Cheng et. al [23] proved that the restricted Delaunay triangulation of an ε-sampling of the skin surface has the closed ball property when ε is small enough. In this section, I will first introduce the definition of the ε-sampling and the properties of its restricted Delaunay triangulation. Then, I describe the restricted union of balls to define an even ε-sampling.

ε-sampling of skin. A finite subset $P \subset F_B$ is an ε-sampling of F_B if every point $x \in F_B$ has a point $p \in P$ such that their distance is at least $\varepsilon\varrho(x)$. In other words, for any point $x \in F_B$, the sphere center at x with a radius $\varepsilon\varrho(x)$ contains at least one sample point. See Figure 4.2 for an example.

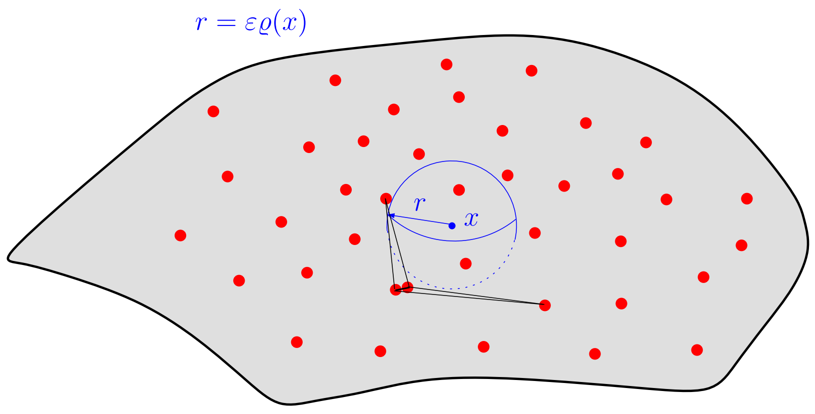


Figure 4.2: Definition of the ε-sampling of a smooth surface.

Homeomorphic Conditions. The restricted Delaunay triangulation is homeomorphic to the surface if it satisfies the closed ball property [50]. This implies that we need a dense enough sample on the surface. For an ε-sampling on the skin with a feasible ε value, its *restricted Delaunay triangulation* is homeomorphic to the skin surface. We refer to the results from Cheng et. al [23].

Thm. 4.2.1 *Homeomorphism Theorem. If P is an ε-sampling of a skin surface F_B with $0 < \varepsilon < 0.279$, the restricted Delaunay triangulation D'_P is homeomorphic to F_B.*

The restricted Delaunay triangulation includes all the Delaunay triangles whose Voronoi edges intersect the skin surface. For each triangle $abc \in D'_P$, see Figure 4.3, its Voronoi edge passes through its circumcenter o and intersects the skin surface at a point z. Moreover, the triangle abc has a small circumradius compared to local length scale at its vertices and the distance between o and z has an upper bound as well. The local length scale of the triangle abc is defined as $\varrho_{abc} = \min\{\varrho(a), \varrho(b), \varrho(c)\}$. I prove the following two lemmas to induce these two properties.

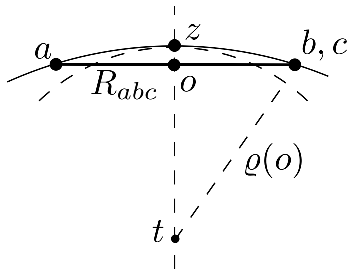


Figure 4.3: A restricted Delaunay triangle abc and its Voronoi edge. o is the circumcenter of triangle abc. z is the intersection of the Voronoi edge with the skin surface.

Lemma 4.2.1 *The cricumradius R_{abc} have an upper bound, namely,*

$$R_{abc} < \frac{\varepsilon}{1-\varepsilon}\varrho_{abc}.$$

PROOF. Assume $\varrho_{abc} = \varrho(a) < \varrho(b), \varrho(c)$. Since z is the intersection of the Voronoi edge of triangle abc with the skin surface, we have $\|z - a\| \leq \varepsilon\varrho_(a)$. According to the curvature variation lemma in Section 2.2.3, we have $\varepsilon\varrho_(a) \leq \varepsilon\varrho(z) + \varepsilon\|z - a\|$. As a result,

$$\|z - a\| \le \frac{\varepsilon}{1 - \varepsilon} \varrho(a).$$

Since the triangle azo is a right angle triangle, we have $R_{abc} \le \|z - a\|$, which implies the claim of the upper bound for R_{abc}. $\boxminus$

Lemma 4.2.2 *The distance between o and z is*

$$\|o - z\| \le \frac{\varepsilon^2}{2} \varrho_{abc}.$$

PROOF. Without loss the generality, let $\varrho_{abc} = \varrho(a) < \varrho(b), \varrho(c)$. According to the definition of the ε-sampling on a skin surface, we have $\|z - a\| \le \varepsilon\varrho(z)$. By curvature variation lemma, we have $\frac{1}{1+\varepsilon}\varrho_{abc} \le \varepsilon\varrho(z)$.

When the vertices a, b and c lie on a sandwishing sphere with radius $\varrho(z) = \frac{1}{1+\varepsilon}\varrho_{abc}$ that passing through z, the distance between z and o is biggest. In this situation, we have $\frac{\|o-z\|}{\|z-a\|} = \frac{\|z-a\|}{2\varrho(z)}$ because the equality of the angle $\angle zao$ and $\angle ota$. Therefore,

$$\|o - z\| \le \frac{\|z - a\|^2}{2\varrho(z)} \le \frac{\varepsilon^2}{2} \varrho_{abc}.$$

$\boxminus$

Lemma 4.2.2 was referred as Circumcenter Lemma. We use these two bounds as the conditions to select candidate surface triangles from the Delaunay triangulation of a partial sampling.

Theorem 4.2.1 tells us that an ε-sampling with small ε value can certify the correctness of the topology of the restricted Delaunay triangulation. However, the restricted Delaunay triangulation may include bad shaped triangles because two sample points in an ε-sampling can be arbitrary close to each other. This leads to short edge length in the restricted Delaunay triangulation, which result in bad shaped triangles. See Figure 4.2 for an example. Next, we introduce the restricted union of balls to define an even ε-sampling that has restricted Delaunay triangulation with quality guarantees.

Restricted Union of Balls. For each point $p \in P$, we define the γ-ball $\hat{p}$ of p as the open ball centered at p with a radius $\gamma\varrho(p)$, in which γ is a positive constant less than 1 and the $\varrho(p)$ is the local length scale at p. A γ-ball is *empty* if no other sample point in P is inside the γ-ball. Each γ-ball intersects the skin surface with a topological disk and the intersection of the skin and a union of a set of γ-balls is called *the restricted union of balls*. The boundary of the restricted union of balls is a set of closed curves consisting of a loop of arcs when the union of balls does not cover the whole surface. A γ-ball that contributes to the boundary of the restricted union of balls is called a *boundary ball*. See Figure 4.4 for an example.

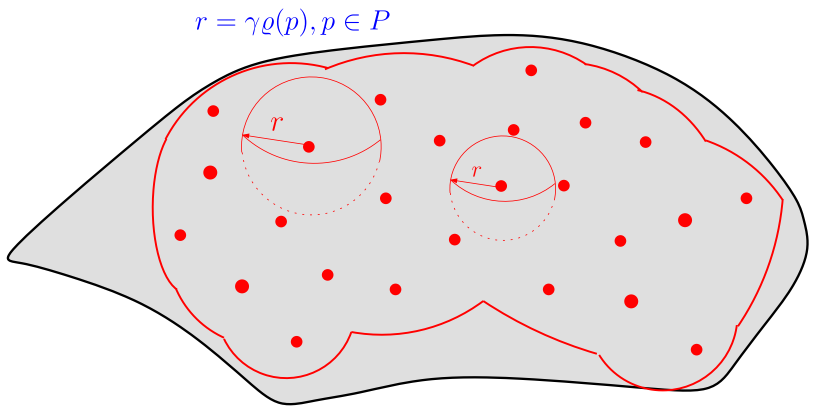


Figure 4.4: The properties of a restricted Delaunay triangle.

We use the restricted union of balls to generate an ε-sampling of the skin surface. We prove that the restricted union of balls without boundaries determines an ε-sampling of the skin surface when the value of γ is small. We have the following theorem.

Thm. 4.2.2 *Sampling Density Theorem. For a sampling $P \subset F_B$, if its restricted union of balls with $\gamma \leq \frac{\varepsilon}{1+\varepsilon}$ covers the skin surface, P is an ε-sampling of the skin surface F_B.*

Proof. Let x be any point on the skin surface. Since the restricted union of balls covers the whole surface, we have at least one γ-ball $\hat{a}$ enclose x. We have the upper bound for γ when x is on the boundary of $\hat{a}$. In this case, $\|x - a\| = \gamma\varrho(a)$.

By definition of the ε-sampling, we require $\|x - a\| \leq \varepsilon\varrho(x)$. Together with the curvature property, we have $\gamma\varrho(a) \leq \varepsilon(\varrho(a) - \gamma\varrho(a))$. That is, $\gamma \leq \frac{\varepsilon}{1+\varepsilon}$.

$\square$

Theorem 4.2.2 implies that we can construct a restricted union of balls with $0 < \gamma < 0.218$ to obtain an ε-sampling with $0 < \varepsilon < 0.279$, which suggests that we can guarantee the surface mesh is homeomorphic to the skin surface.

If we require each γ-ball is empty, we can guarantee the length of an edge ab in the surface mesh have a lower bound, that is,

$$\|a - b\| \leq \gamma\varrho_{ab},$$

in which the local length scale of the edge ab is defined as $\varrho_{ab} = \max\{\varrho(a), \varrho(b)\}$. Together with the Lemma 4.2.1, we can guarantee a lower bound on the minimal angle of the surface mesh, that is, $arcsin(\frac{\gamma}{2\varepsilon}(1-\varepsilon))$. This suggests that smaller γ and ε mean better mesh quality if we ensure that they satisfy the upper bound. We can choose the values of the γ and ε as small as possible. However, this would result in more surface triangles. We should keep a balance between the mesh quality and mesh size. For example, we chose $\gamma = 0.15$ and $\varepsilon = 0.18$ and achieved a lower bound of 20° on the minimum angle.

4.3 Components Computation

A skin surface consists of one or more disjoint components. The algorithm in this chapter constructs the triangulation of the skin surface F_B by meshing each component individually. A component of a skin surface either corresponds to connected components of dual complex K_B or a void in K_B because the skin body, *body*(B), is homotopic to the underlying space of the dual complex K_B [38]. Thus, we can compute all the connected components of a skin surface by computing the connected components and voids in the dual complex because the dual complex K_B is a simplicial complex, which support fast combinatorial algorithm for computational purpose. It is trivial to compute the connected components in a simplicial complex. I will focus on the computation of the voids in the dual complex K_B.

The dual complex K_B of B is a subcomplex of the weighted Delaunay triangulation D_B. A void in K_B is the boundary of a collection of tetrahedra in the complement of K_B in terms of D_B, that is, $D_B - K_B$. I use the incremental algorithm for Betti numbers of simplical complex on the 3-sphere [34] to compute the voids in the dual complex. In this section, I introduce the filtration of the subcomplex of D_B firstly. Then, I describe the incremental algorithm for the voids computation. Finally, I introduce data structure used in the computation.

Filtration. A filtration is sequence of simplicial complex so that each complex is a proper subcomplex of its successor. We can build a filtration of the subcomplex in D_B under the growth model we introduced in Section 2.1.4, namely, the *alpha filtration*, which facilitates efficient constructions of the voids in K_B.

Recall that we have four types of simplices in the Delaunay triangulation D_B, namely, vertices, edges, triangles and tetrahedra. Each simplex δ_X has a birth time ζ_X that indicates the α value when δ_X enters the α-complex when the α is growing from $-\infty$ to ∞. Assume the number of simplex in D_B is t and we can order all the simplex in D_B with a sequence $\langle \delta_1, \delta_2, \delta_3 \cdots, \delta_t \rangle$ according to their birth time with the following rule: For any $\delta_i = \delta_X$ and $\delta_j = \delta_Y$, $i \leq j$ if (i) $\zeta_X \leq \zeta_Y$ or, (ii) $\zeta_X = \zeta_Y$ and $\text{card}(X) < \text{card}(Y)$. The sequence $\langle \delta_1, \delta_2, \delta_3 \cdots, \delta_t \rangle$ is called a *filter*.

The alpha filtration is

$$\emptyset = K_0 \subset K_1 \subset K_2 \cdots K_m = D_B,$$

where $K_i = \{\delta_j | 1 \leq j \leq i\}$ and $K_i - K_{i-1} = \delta_i$ for all $1 \leq i \leq m$. Figure 4.5 illustrates an example filtration.

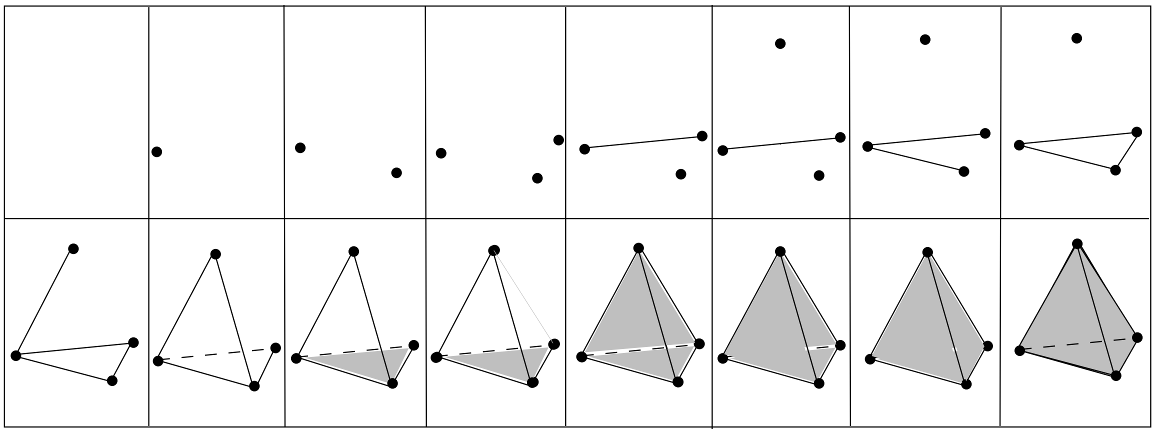


Figure 4.5: A filtration from empty set to a tetrahedron.

Voids Computation. We represent a void in K_B with a collection of tetrahedra in $D_B - K_B$. To compute such a collection of tetrahedra, we scan the alpha filtration backward. In particular, we store the alpha filter in a linear array T. For the number of simplices in K_B, m, all simplices $\delta_i, i \leq m$ belong to the K_B and the simplices after position m belong to $D_B - K_B$. Starting from the last tetrahedron in the array T, we scan the triangles and tetrahedra in the array until the index m to compute the voids. The voids computation is illustrated as following.

Algorithm 4.1 VoidsComputation()

```
1: for i= t downto m+1 do
2:     $\delta$ = T[i];
3:     if $\delta$ is a tetrahedron then
4:         Create a set and add $\delta$ to the set;
5:     else if $\delta$ is a triangle then
6:         Find the sets that contains the tetrahedra with $\delta$ as a face;
7:         if There are two different sets are found then
8:             Merge the two sets;
9:         end if
10:    end if
11: end for
```

The computation simulates the birth and growth of the voids as the α value decreases from ∞ to 0. When the algorithm terminates, we have a number of sets consisting of tetrahedron. Each set represents a void except for a special set that represents the unbounded parts of the complement of K_B [34].

Union-Find Data Structure. The upper algorithm is supported by the union-find data structure, which represents a collection of elements partitioned into pairwise disjoint sets. The data structure supports the following types of operations.

- Add(e) : Add e as the only element of a new set $\{e\}$.
- Find(e): Determine and return the sets that contain e.
- Union(A, B): Merge the two sets A and B by their union.

In the algorithm 4.1, the elements are the tetrahedra and triangles in the array T and each set represents a void. Since the union-find data structure support efficient implementations, we can compute the voids in the dual complex very fast, which also implies that we can compute all the connected components in a skin surface efficiently. Next, we introduce the algorithm that generates quality meshes for each component using restricted union of balls.

4.4 Algorithm

I describe the algorithm triangulating each connected component in a skin surface in this section. I will first give an overview of the algorithm. A few details in the algorithm are described later.

4.4.1 Overview

For each component, we start from two seed triangles and insert new points along the boundary of the restricted union of balls incrementally. Each newly inserted point is located outside the current restricted union of balls. After a new point is inserted, we compute the Delaunay triangulation and extract the small restricted Delaunay triangles as the candidates for the surface triangulation. At the same time, the restricted union of balls expands along the skin surface and the boundary is updated until the whole surface is covered. The algorithm terminates when the restricted union of balls covers the whole surface. Figure 4.6 shows the initial construction of the restricted union of balls from two seed triangles.

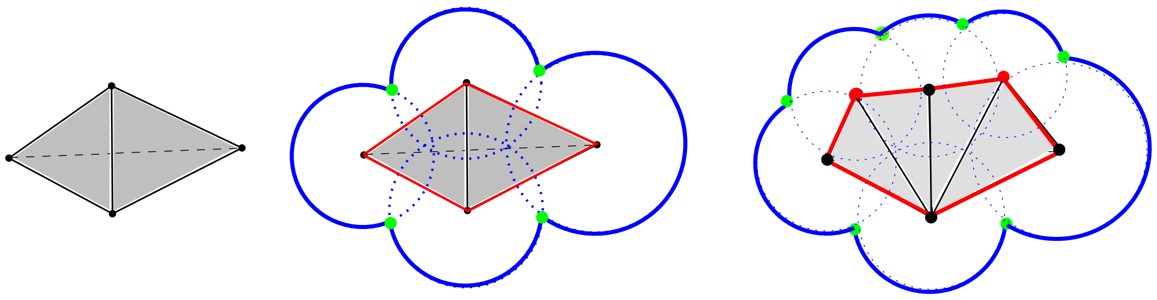


Figure 4.6: The initial construction of the restricted union of balls.

Denote the current sampling point set as P_i and Delaunay triangulation of P_i as D_i. We choose all the small restricted Delaunay triangles in D_i as the *candidate surface triangles*, denoted as $S_i \subset D_i$. Ideally, S_i could be a piecewise 2-manifold with boundary that is an exact subset of the final surface mesh. However, we cannot achieve this from a partial sampling and the candidate surface triangles S_i is a superset of the 2-manifold, which may include some false surface triangles resulting in tetrahedra or non-manifold elements. These false surface triangles can be cleaned up when we obtain an ε-sampling.

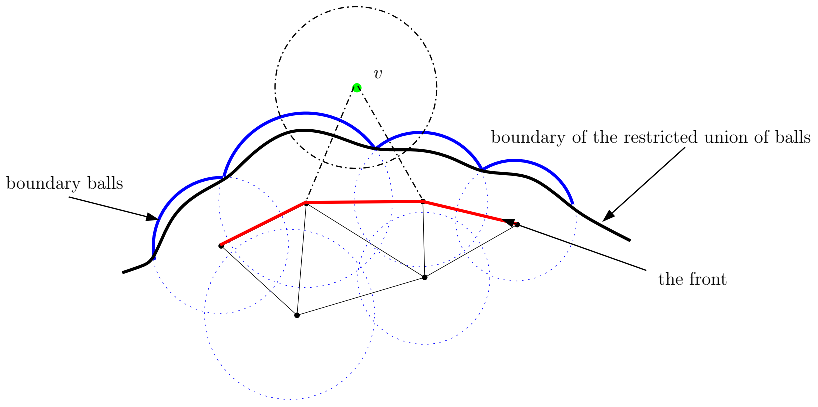


Figure 4.7: The vertex insertion in the algorithm.

With P_i, we insert a new point v to get P_{i+1}. Figure 4.7 illustrates a scenario of a vertex insertion. Since the point v must be outside of the current restricted union of balls, we identify the boundary balls to locate v. The boundary balls are found by using a subset of the Delaunay edges in S_i, namely, the $front$. Each front edge is either a dangling edge in S_i or an edge shared by two candidate triangles whose normals form an angle larger than 90°. The collection of all the γ-balls at the vertices of the front edges includes all the boundary balls and a small number of interior γ-balls. We can differentiate these interior balls in our point placement methods and locate the new point v on the restricted Voronoi edge of a front edge whose vertices γ-balls are boundary balls. Then, we maintain the Delaunay triangulation of P_{i+1} and extract new candidate surface triangles. As a result, the collection of candidate surface triangles grows from S_i to S_{i+1} and the front advances to the unmeshed region. We store the $front$ in a queue Q and iteratively apply the above procedure until the front Q is empty, that is, the restricted union of balls cover the whole surface. The pseudo code of the algorithm is illustrated as following.

Algorithm 4.2 DeloneSkinMesh()

```
1: while The boundary of restricted union of balls is not empty do
2:     Find a front edge $ab$;
3:     Locate a new point $v$ according to $ab$;
4:     Compute the Delaunay triangulation of $P_i \cup \{v\}$;
5:     Extract the candidate surface triangles;
6:     Update the front;
7: end while
8: Clean up the non-restricted Delaunay triangles from $S$;
```

The algorithm consists of four main steps, namely, locating a new vertex, updating the Delaunay triangulation, extracting candidate surface triangles and updating the front. When the restricted union of balls covers the surface and no more points can be inserted, we get an ε-sampling and the collection of candidate triangles includes the restricted Delaunay triangulation of the ε-sampling. We clean up the non-restricted Delaunay triangles and get the final surface mesh. Next, we discuss these steps in details.

4.4.2 Point Placement

Assume that we have a front edge ab whose γ-balls $\hat{a}$ and $\hat{b}$ are boundary balls. We aim to locate a new sample point v satisfying the following requirements:

- the point v should be outside the restricted union of balls;
- the γ-ball $\hat{v}$ should be free of any sample point in P_i;
- the γ-ball $\hat{v}$ should intersect the boundary balls $\hat{a}$ and $\hat{b}$ deeper than tangentially;
- the point v will form at least one candidate surface triangles with the points in P_i.

The first two requirements maintain the empty property of the γ-balls, which avoids short edges in the mesh and implies that the algorithm will terminate. The third requirement is to ensure the restricted union of balls covers the whole surface once the algorithm terminates. The last one aims to maintain a valid front to guide our future vertex insertion. These requirements imply that the distance from the point v to a and b should be larger than $\gamma \varrho_{ab}$ but not too far from a and b. This suggests that the points along the intersection between the skin and the boundary balls $\hat{a}$ and $\hat{b}$ would be a good choice. We argue that there is always a space to find a point v satisfying these requirements before the algorithm terminates. In particular, we can locate the point v on the restricted Voronoi edge of the front edge ab. See Figure 4.8 as an example.

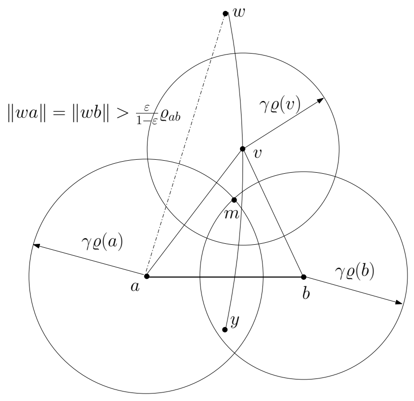


Figure 4.8: Locate the new point v correspond to a front edge ab.

In Figure 4.8, ab is a front edge and its restricted Voronoi edge is the curve wy, which is the intersection of the skin and the Voronoi polygon of ab with respect to the points in P_i. The solid line circles represent the γ-balls and the point m is the intersection point of the skin surface and the boundary of $\hat{a}$ and $\hat{b}$ that is outside the restricted union of balls. The point v is the point we intend to insert and edge va and vb are the Delaunay edges of D_{i+1} after v is inserted. We demonstrate that point v can be found on the curve wy around the point m. Since ab is a front edge, the endpoint w is on the Voronoi edge of a Delaunay triangle abx in D_i and abx is not a candidate surface triangle. Thus, the circumradius R_{abx} of the triangle abx is greater than $\frac{\varepsilon}{1-\varepsilon}\varrho_{abx}$. It implies that $\|wa\| = \|wb\| > \frac{\varepsilon}{1-\varepsilon}\varrho_{abx}$, which means the point w is outside $\hat{a} \cup \hat{b}$ and far from the corner point m. As a result, we can find the point v starting from m toward w along the curve wy.

The computation of the intersection curve wy between the skin and the Voronoi polygon of edge ab is costly. Therefore, we use the projection of wy on the tangent plane of the point a, namely, $w'y'$, to locate the point v. We denote the projection of m on the line $w'y'$ as m' and $w'y'$ is a line segments passing through the middle point t of a and b. We find a point v' on $w'y'$ start from t with a length slightly longer than tm'. For instance, $\|tv'\| = \frac{1+\sqrt{3}}{2}\|tm'\|$ is a feasible choice in practice. We get the point v by projecting v' to the skin surface.

In the case where the vertices γ-balls of front edge ab is not boundary balls, the point v we compute using the previous procedure must be inside the restricted union of balls. We simply discard the point v and cancel the candidate surface triangles attached to the edge ab. Next, we compute the Delaunay triangulation, D_{i+1}, of $P_i \cup \{v\}$.

4.4.3 Computation of Delaunay Triangulation

We adapt the incremental flip algorithm to construct the Delaunay triangulation D_{i+1} efficiently. The incremental flip algorithm was initially proposed by Lawson [72]. The basic idea of the algorithm is the following. Let P be a set of n points in $\mathbb{R}^3$, $4 < i < n$ and assume that the Delaunay triangulation of the first i points in P is already constructed, called D_i. Add the $(i+1)$-th point to triangulation and restore the Delaunayhood by flipping, this results in D_{i+1}. Repeat this process until $i = n$. A crucial step in the algorithm is the point location, which occurs when a new point is added in to the triangulation. A directed acyclic graph (DAG) with the history of all performed flips were used to speed up the point location. However, the DAG structure results in large memory usage and complicated implementation. In our algorithm, we employ a straightforward yet fast way to locate point by taking the advantage of our point sampling strategy.

Since the newly inserted point v corresponds to a front edge ab and is not far from a and b, we simply search the tetrahedra in a certain region around ab. Particularly, we choose a Delaunay triangle abd in the star of edge ab whose normal has minimal difference with the normal of point v at the skin surface as a starting point. The point v only belongs to one of the half spaces divided by the plane pass through triangle abd, denote as H. If the triangle abd is a facet on the convex hull and there is no points in P_i belonging to the half space H, the point v would be on the convex hull also and we return the artificial tetrahedron connecting abd with the point at infinite. Otherwise, there must be a tetrahedron enclosing v and we can find it by walking through the triangles enclosed by a sphere centered at a and with a radius of 1.5 times of the length of edge va along the direction toward v. In general, the point v is either a convex hull point or is enclosed by a tetrahedron attached to the start point triangle. That is, we only need a constant time to locate the tetrahedron enclosing v, which accelerate the construction of the Delaunay triangulation very much. Then, we connect v to the vertices of the tetrahedron and perform flips to restore the Delaunay property of the triangulation. Our implementation is based on the results of alpha shape software [1, 49].

After the Delaunay triangulation D_{i+1} is accomplished, we extract the candidate surface triangles. The Delaunay triangulation D_{i+1} differs from D_i with the star of v, which is a set of tetrahedra that include v as one of their vertices. Therefore, we can only consider the triangles in the star of v for labeling new candidate surface triangles.

4.4.4 Extraction of Candidate Surface Triangles

We select candidate surface triangles according to two conditions, namely, the small circumradius condition and restricted Delaunay condition. By"small" we mean the circumradius R_{abc} of a Delaunay triangle abc is smaller than $\frac{\varepsilon}{1-\varepsilon}\varrho_{abc}$. For the restricted Delaunay property, we not only require the Voronoi edge V_{abc} intersects the skin surface but also need the distance between the intersection point z and the circumcenter o is less than $\frac{\varepsilon^2}{2}\varrho_{abc}$. The following pseudo code shows the procedure $ExtractCandidateTringles()$.

Algorithm 4.3 ExtractCandidateTriangles()

```
1: unmark all the triangles in $Star(v)$
2: while there is an unmarked triangle $abc \in Star(v)$ do
3:   if $R_{abc} < \frac{\varepsilon}{1-\varepsilon}\varrho_{abc}$ then
4:     if $V_{abc}$ intersect the skin surface with a closest point $z$ and $\|oz\| < \frac{\varepsilon^2}{2}\varrho_{abc}$
       then
5:       mark $abc$ as a candidate surface triangle;
6:     else
7:       mark $abc$ as a non candidate surface triangle;
8:     end if
9:   else
10:    mark $abc$ as a non candidate surface triangle;
11:    for the edge $xy \in \{ab, bc, ac\}$ do
12:      if $\|xy\| > \frac{2\varepsilon}{1-\varepsilon}\varrho_{abc}$ then
13:        mark all the triangles in $Star(xy)$ as non candidate surface triangles;
14:      end if
15:    end for
16:  end if
17: end while
```

The small circumradius condition and restricted Delaunay condition ensure that the candidate surface triangles include all the restricted Delaunay triangles in the final restricted Delaunay triangulation of the ε-sampling. This argument is based on the following two observations. First, the small Delaunay triangles would be always Delaunay if we did not insert any point into its smallest circumsphere. It implies that the points sampled in the future will not invalidate most existing candidate surface triangles. Second, the restricted Voronoi edge of a Delaunay triangle can only become shorter after more points are sampled, which means that a restricted Delaunay triangle in a partial mesh could become a non-restricted Delaunay triangle but a non-restricted Delaunay triangle would never be a restricted Delaunay later.

We can accelerate the candidate surface extraction procedure by checking the edge length of a large triangle. Since the longest edge of a triangle is as long as 2 times of its circumradius, we can distinguish whether a Delaunay triangle is not a candidate surface triangle if its edge is too long.

Finally, we update the front from the new candidate surface triangles by checking each edge of a candidate triangle. If the edge is a dangling edge or shared by two triangles with large normal angles, then we put the edge to the queue Q.

If the restricted union of balls covers the whole surface and no more points can be added, the sample points P is an ε-sampling of the skin surface. We walk through all the candidate surface triangles and clean up all the non-restricted Delaunay triangles. The remaining triangles form the restricted Delaunay triangulation of the ε-sampling P, which is a quality surface mesh approximating the skin surface.

4.5 Results

I implemented the algorithm on the PC platform with C++ and OpenGL as the graphics library. I partially reuse prior software on Alpha shape [1] and Betti numbers [34]. I triangulate a few large molecular skin models of proteins, especially several molecules that the algorithm in Chapter 3 fails to generate the mesh because of the noisy critical points in the Morse-Smale Complexes.

Figure 4.1 shows the molecular skin model of a protease molecule of the Human Immunodeficiency Virus (HIV). Our algorithm triangulates this surface and produce a mesh with a minimum angle 20.35°. Since the surface of the molecule has a very bumpy appearance, our previous algorithm failed to triangulate this surface [25]. In contrast to this, our current algorithm can handle the bumpy surface robustly. This can be explained by two factors: first, we did not use the Morse-Smale complex in this algorithm so that there are no accumulation of numerical error involved in the computation; second, the robust and stable implementation of the alpha shape software relieve us from the robustness worries associated with the Delaunay triangulation computation.

Figures 4.9 and 4.10 show the molecular skin model of two molecules we randomly choose from the protein data bank [2]. These results demonstrate that our meshing algorithm generates a correct and precise representation of the surfaces of molecules. We verify this by comparing the Betti numbers of the alpha shape and that of the surface. The topological features such as genus and tunnels are preserved and detail geometric feature like depressions and protrusions are approximated accurately. For example, Figure 4.9(b) shows the magnified view of a small genus and Figure 4.10(b) illustrates the magnified view of a cavity on the surface. The homeomorphism between the mesh and the original surface and the accurate approximation are due to the ε-sampling generated by the restricted union of balls. This also supports our arguments that the molecular skin model is a better geometric model for the molecules such as proteins and DNAs because it can enable us to achieve these two goals.

molecular name	no. triangles in the mesh	minimum angle in the mesh	computing time
$200D(Figure4.9)$	65,162	19.28°	00:01:35
$1FG1(Figure4.10)$	94,390	20.82°	00:02:41
$HIV2(Figure4.1)$	226,758	20.35°	00:15:43

Table 4.1: Performance of the meshing algorithm using restricted union of balls.

Table 4.1 lists the statistics of the running time and mesh quality of the surface meshes illustrated in this paper. All these experiments were run on a Pentium 4 PC. These results indicate that our implementation performs robustly and achieves good mesh quality with reasonable speed.

4.6 Summary

This chapter describes an incremental mesh generation algorithm using restricted union of balls for triangulating the molecular skin surfaces. The restricted union of balls is the intersection of the skin surface and the union of a set of empty open balls centered at the surface sample points. By judiciously choosing the radius of the balls to adapt to the surface curvature, we obtain an even ε-sampling and the surface mesh is the restricted Delaunay triangulation of the sampling, which is guaranteed to be homeomorphic to the original surface. At the same time, we achieve guaranteed mesh quality and reasonable efficiency.

Compared with the sweeping skin meshing algorithm, our current meshing algorithm performed more robustly but had a lower efficiency because it computes the Delaunay triangulation of all the sample points on the surface. Although we improve the efficiency of locating the tetrahedron in the incremental construction of the Delaunay triangulation in a fast way, the edge flipping is still costly since the complexity of the Delaunay triangulation of surface sample points can be $O(n^2)$, as shown by Jeff Erickson[54]. Actually, any surface meshing algorithm using Delaunay triangulation of the sample points would face this problem and perform even worse because they use the DAG structure to locate the new vertex.

The mesh quality we achieved is worse than the previous methods, namely, the sweeping skin meshing algorithm and dynamic skin triangulation algorithm in a small scale. The main reason is that we do not use the costly refinement operations to improve the mesh quality. We can achieve a better quality by choosing smaller γ values. However, it would result in a large number of triangles in the mesh. Applying the refinement operations on our result would be an option to achieve better qualities.

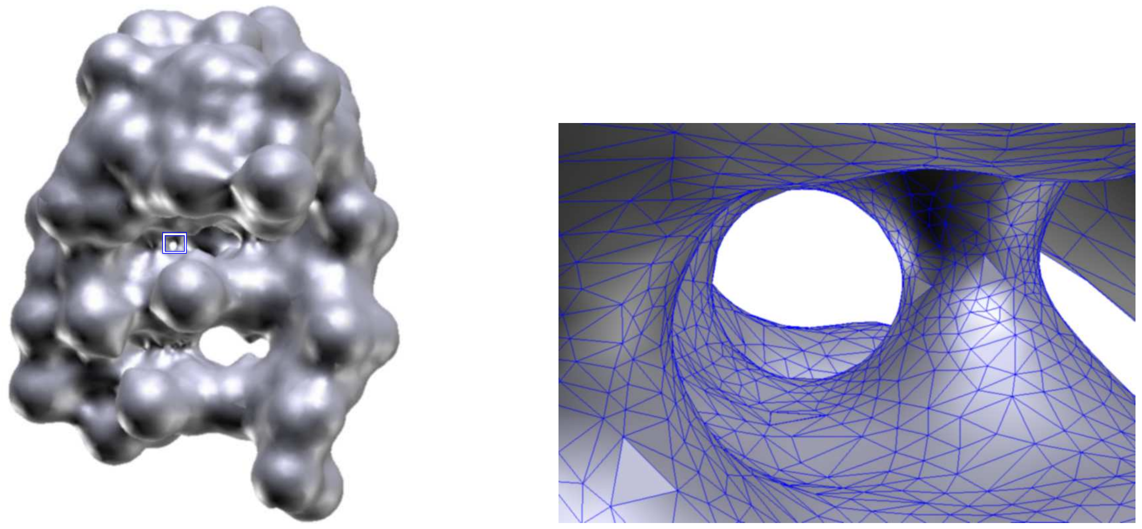

(a) (b)

Figure 4.9: The molecular skin model of the molecule with PID:200D. Figure (a) shows its surface model and (b) illustrates the zoomed mesh details of a small genus in the box of (a).

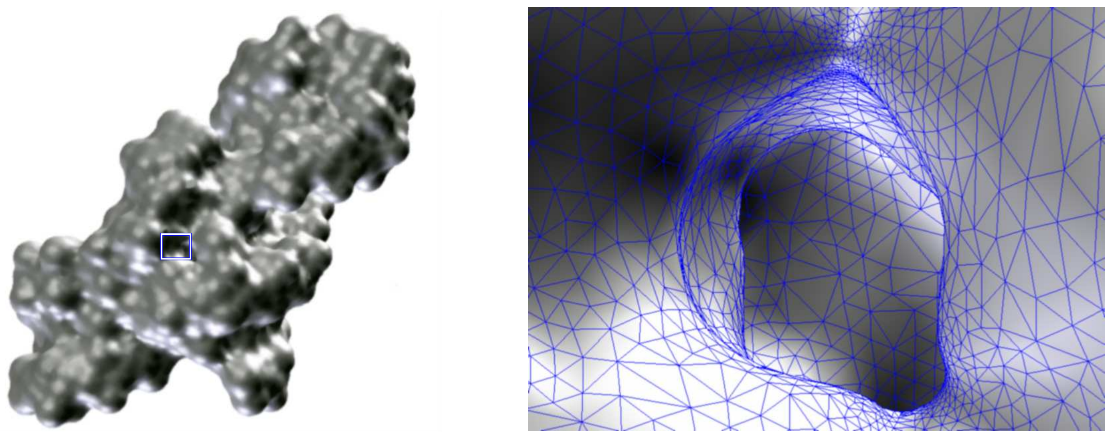

(a) (b)

Figure 4.10: The molecular skin model of the molecule with PID:1FG1. Figure (a) shows its surface model and (b) illustrates the zoomed mesh details of a cavity in the box of (a).

Chapter 5

Quality Tetrahedral Mesh Generation for the Skin Body

This chapter develops mesh generation algorithm for building quality tetrahedral meshes of the volumes bounded by the skin surface, which are desirable for the electrostatic computation of macromolecules. Specifically, the algorithm generates tetrahedral meshes that are the subsets of the Delaunay triangulation. By taking the advantages of the skin surface meshing results in Chapter 3 and Chapter 4, the algorithm starts by building a coarse tetrahedral mesh for the skin body. Next, the algorithm applies the Delaunay refinement to improve tetrahedral quality with a priority parameterized by the value of the distance function defined by the surface. Together with the nice properties of the input surface mesh, we achieve an upper bound on the radius-edge ratio of the tetrahedral mesh after the refinement. Finally, we apply the sliver exudation algorithm to remove "slivers". The algorithm terminates with guarantees on the tetrahedral quality and an accurate approximation of the original surface boundary.

Section 1 gives a brief introduction of the numerical methods for computing the electrostatic property and defines the quality tetrahedral meshes. Section 2 reviews the Delaunay refinement methods and summarizes the main challenges for applying the methods to reach our goal. Section 3 describes our meshing algorithm and analyzes its behavior. Experimental results are illustrated in Section 4 and the chapter is concluded in Section 5.

5.1 Numerical Methods and Mesh Quality

Numerical methods approximate the solution of partial differential equations (PDE) which model the physical phenomena in science and engineering such as heat transfer, quantum mechanics and electrostatic property of the molecules. These methods usually decompose the problem domain into simple pieces, namely, *elements*, and approximate the solution of the PDE on each element. The shape of the element is essential to the accuracy and stability of the approximation. In this section, I will give a brief introduction of the *finite element methods* (FEM) for computing the electrostatic property to motivate the study of tetrahedral mesh of the skin body. Then, I introduce the measures of the tetrahedral mesh quality to define quality meshes for the FEM methods.

Electrostatics Computation. Electrostatics potential is one of the fundamental energy terms to model a molecular system. It defines the potential energy at a particular location near a molecule created by the system of molecular charges. The study of the electrostatic potential within a molecule or their interactions among different molecules is necessary to investigate the protein folding and protein-protein interactions. Thus, modeling and computation of the electrostatic of molecules have become a central topic in the molecular modeling study [10]. Several methods have been proposed to compute the electrostatics potential of molecules. The simplest approach is to apply the Coulomb's law with a fixed dielectric constant. This approach neglects the presence of a dielectric interface between the molecular interior and the surrounding solvent. A relatively rigorous framework is to use the continuum approach that models a molecule as a domain with low dielectric constant inside a smooth surface boundary embedded in a continuous medium of solvent with high dielectric constant [73].

The Poisson-Boltzmann equation(PBE) is one of the most popular continuum approach to model the electrostatic of large molecules. In this method, the electrostatic potential $\phi(X)$ at a location $X \in \mathbb{R}^3$ is modeled with the following the Poisson equation

$$-\nabla \cdot \epsilon(X)\nabla\phi(X) = \rho(X),$$

in which $\epsilon(X)$ is a spatially varying dielectric coefficient, and $\rho(X)$ is the charge distribution in the molecules. In particular, the dielectric coefficient $\epsilon(X)$ jumps by one or two orders of magnitude at the interface between the inside the molecules and the solvent. It results in the solution of the equation is quite sensitive to the boundary definition of the molecular model [10].

Numerical methods are necessary to solve the PBE because the equation is a non-linear partial differential equation of second order and analytical solutions are not available. There are three types of numerical methods for solving the partial differential equation including finite difference methods, boundary element methods and finite element methods [59, 63, 75, 80, 84]. Finite difference methods employ regular grid to subdivide the domain encompassing the molecule and its surroundings so that the value of the solution can be evaluated at each grid point. However, the Cartesian nature of the grids makes it impossible to locally increase the accuracy of the solution in a specific region without increasing the resolution of the entire grid [80]. The boundary element methods are limited to solve the simplified form of the

PBE [75]. On the other hand, finite element methods offer the ability to solve the PBE with controlled accuracy in specific regions of the problem domain [20]. A finite element solver usually partitions the domain in three dimensions into a collection of tetrahedra, namely, the finite element mesh. Then, the PBE is approximated by basis functions on each tetrahedron and the solution is derived from a system of equations formed by the basis functions. Specifically, the finite element method constructs a 4×4 matrix K_τ for each tetrahedron τ, namely, the *element stiffness matrix.* Let the vertices of τ be i, j, k, l. Denote ℓ_{kl} as the edge length from k to l and θ_{kl} denote the dihedral angle at the edge. The *(i,j)*-th entry of the matrix K_τ equals to

$$K_\tau(i,j) = -\ell_{kl} \cot \theta_{kl},$$

and the diagonal entries are the inverse of the sum of all the entries in the row. It is clear that when $\theta \rightarrow 0°$ or $\theta \rightarrow 180°$, then $\cot \theta \rightarrow \infty$. Large entries in the matrix can result in large eigenvalues of the matrix, which affect the solution time and the solution accuracy of the system of equations [91]. Therefore, the shape of tetrahedra decomposing the molecular volume would be critical to solve the PBE accurately and efficiently using finite element methods.

In summary, a quality volumetric mesh of molecules that conforms to its boundary is desirable for the computation of the molecular electrostatic by solving the PBE with finite element methods. However, it is still challenging to construct quality tetrahedral meshes conforming to the boundary of the molecular model. Most of the previous works used regular grids [84] or adaptive grids based on octree subdivisions [99]. The mesh elements in these methods have a biased alignment to the axis and cannot conform to the boundary surface accurately because the the resolution of the grids can not be infinite fine. Delaunay meshes have no such problems and support efficient construction algorithms with quality guarantees [90]. The algorithm developed in this chapter can generate quality Delaunay tetrahedral meshes for macromolecules. Next, we describe the measures that define a quality Delaunay tetrahedral mesh.

Tetrahedral Quality. A quality measure for tetrahedra is used to identify the poor shape tetrahedra, that is, the tetrahedra with small or large dihedral angles. For a Delaunay tetrahedral mesh, the *circumradius to shortest edge ratio* of a tetrahedron is the most natural and elegant measure [78]. The *circumsphere* of a tetrahedron is the unique sphere that passes through all its vertices. The center and the radius of the circumsphere are referred as the *circumcenter* and *circumradius* of the tetrahedron respectively. We denote r as the circumradius and l as the shortest edge of a tetrahedron τ respectively. Thus, the length ratio of the circumradius of a tetrahedron and the shortest edge, namely, the radius-edge ratio, is denotes as $\frac{r}{l}$. We call a tetrahedron τ is a *skinny* tetrahedron if its radius-edge ratio is larger than a constant $c > 1$, that is, $\frac{r}{l} \geq c$. Otherwise, we call the tetrahedron τ has *ratio property c*. A tetrahedral mesh K has the ratio property for a constant c means that each $\tau \in K$ has ratio property c. Most poor shape tetrahedra are characterized by their radius-edge ratio. That is, their circumradius are much larger than their shortest edge and they have large radius-edge ratio. Figure 5.1 (a), (b) and (c) illustrates the examples of skinny tetrahedra.

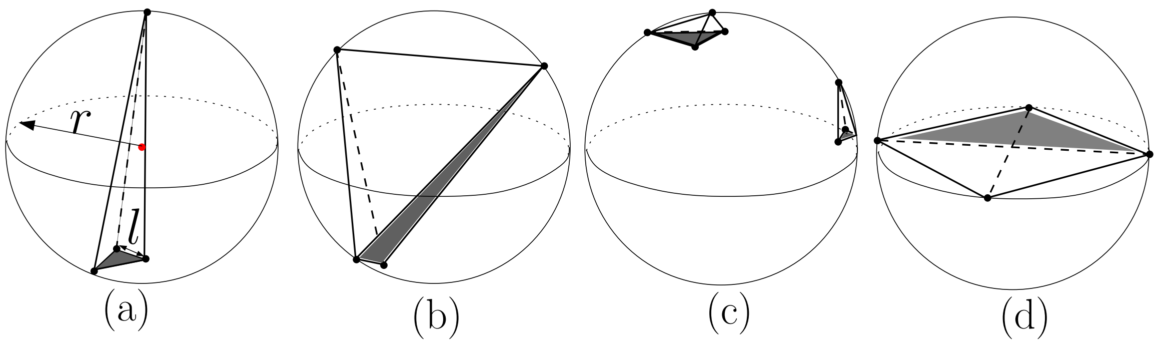


Figure 5.1: A classification of the bad shape tetrahedra.

However, a tetrahedral mesh with bounded radius-edge ratio is not entirely free of tetrahedra with bad angles. The *sliver* is the only type of bad shape tetrahedra with bounded radius-edge ratio. The canonical sliver is formed by arranging four equally spaced vertices around the equator of a sphere, then perturbing one of the vertices slightly off the equator. See Figure 5.1 (d) for an example. The sliver can have small radius-edge ratio as low as $\frac{\sqrt{2}}{2}$ but the dihedral angles can approach to $0°$ and $180°$. In this chapter, a tetrahedron is called a *sliver* if it has the ratio property c and its minimum dihedral angle is smaller than a constant ς. The goal of the meshing algorithm in this chapter is to generate tetrahedral meshes for the skin volume with ratio property and free of slivers. In particular, the algorithm use Delaunay refinement methods to construct a tetrahedral mesh with ratio property, then remove the slivers with a post-processing procedure. Next, I introduce the Delaunay refinement method and summarize the main challenges in applying the method to generate quality meshes for the skin body.

5.2 Delaunay Refinement

Delaunay refinement methods generate quality tetrahedral meshes by placing new mesh vertices carefully in the domain and maintaining the Delaunay triangulation until all mesh elements meet the quality constraints. They performed excellently in both theory and practice because of the elegant mathematical properties of the Delaunay triangulation and efficient construction algorithm. In this section, I will review the key idea behind the Delaunay refinement and the previous mesh generation algorithms using Delaunay refinement. Then, I summarize the challenges in adapting the method to generate quality meshes for the volume bounded by the skin surface.

The Key Idea. The central operation of all the Delaunay refinement algorithms in three dimensions is to insert a vertex at the circumcenter of a poor shape tetrahedron followed by edge flipping to maintain the Delaunay property. The vertex insertion kills the poor shape tetrahedron because its circumsphere is no longer empty. By iteratively applying the operation, the algorithm will eventually eliminate all the poor shape tetrahedra or run forever. Fortunately, Delauany refinement algorithms are always guaranteed to terminate if the poor shape tetrahedra are defined as the skinny tetrahedra with radius-edge ratio greater than the constant $c > 1$. The reason is that the newly created edges after the insertion of the circumcenter of a skinny tetrahedron is at least as long as the shortest edge in the tetrahedron. See Figure 5.2 for an example in two dimensions. All the new edges created by the insertion of t connected to t. Because t is the circumcenter of the triangle abc and there are no other vertices in the circumsphere of abc before t is inserted, all the new edges are at least a long as the circumradius of abc. Since the circumradius to shortest edge ratio of the triangle abc is greater than c, so every new edge is at least c times the shortest edge of abc.

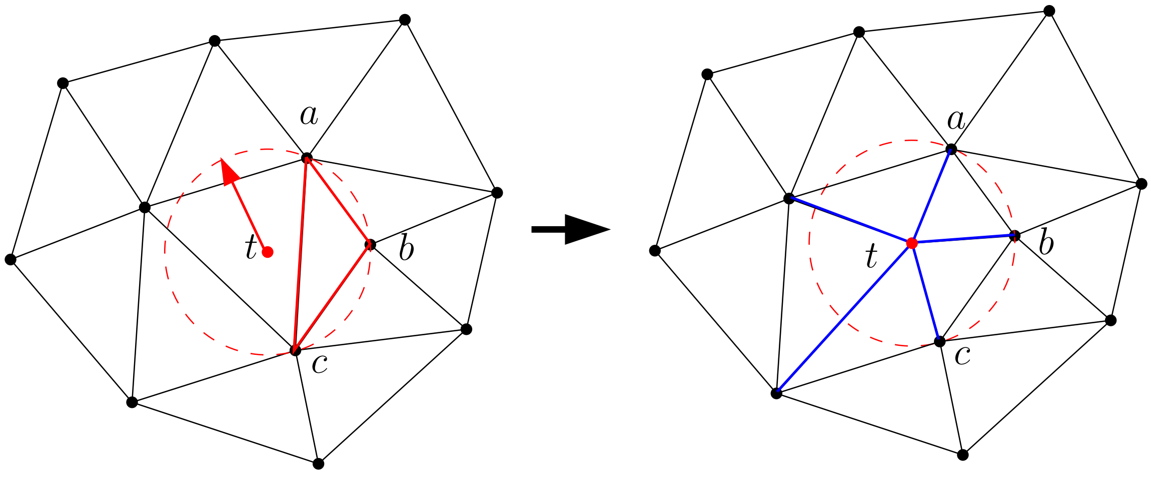


Figure 5.2: The insertion of the circumcenter of a poor shape triangle. The Delaunay property is maintained and all the new edges have length at least c times the shortest edge of triangle abc.

Boundary Recovery. Another important issue in Delaunay refinement algorithms is the boundary recovery when they are applied to a domain specified by a set of polygonal faces, line segments and vertices. In general, the boundary faces and segment are not in the initial Delaunay triangulation of all the boundary vertices. In order to force these boundary faces and segments to appear in the final Delaunay mesh, additional vertices lying on the boundary are necessary to be inserted. For example, the algorithm can split the boundary segments into subsegments by inserting the middle point of the segment once its diameter sphere is not empty. However, an acute input angle on the boundary may result in infinite loops in such a boundary recovery procedure. See Figure 5.3 for an example. Because the triangle contains the vertex a is always a poor shape triangle and its circumcenter lies in the diameter sphere of the subsegments, the splitting will run forever.

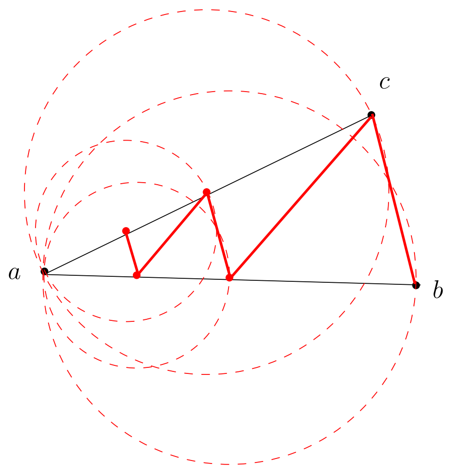


Figure 5.3: The boundary recovery fails when there is an acute input angle.

The Delaunay refinement methods have been studied widely in both two and three dimensions. In three dimensions, most of the previous work concentrated on the meshing the domain bounded by piecewise linear complex (PLC). In the following, I review the previous work to identify the main challenges for my application of the Delaunay refinement methods.

Challenges. Although significant progress has been made in the Delaunay refinement meshing algorithm in three dimensions, it is still challenging to apply the method to meshing the domains bounded by smooth surfaces. Shewchuk [90] used the Delaunay refinement to generate tetrahedral meshes for 3D domains bounded by a piecewise linear complex. The algorithm eliminates poor quality tetrahedra by iteratively inserting their circumcenters. Consequently, the resulting mesh achieve an upper bound on the radius-edge ratio, which is the ratio of the circumradius of a tetrahedron with its shortest edge. However, the algorithm requires the PLC domain has no acute input angles. This problem was addressed recently by Cheng et al. [29], in which the algorithm generates Delaunay meshes for polyhedra with small input angles with guarantees on the radius-edge ratio, except for the tetrahedra near the small input angles. Bounded radius-edge ratio eliminates all kinds of bad shape tetrahedra except slivers. Cheng et al. [28] introduced sliver exudation algorithm by assigning weight to the mesh vertex such that the resulting weighted Delaunay triangulation is sliver free. Subsequently, Edelbrunner et al. [47] perturb the points to clean up the slivers in the mesh. However, both sliver removal algorithms cannot handle the boundaries because they only applied to periodic sets. Recently, Cheng and Dey [27] combined Delaunay refinement with sliver exudation to construct quality tetrahedra meshes for polyhedra without acute input angles.

To conclude, the application of the Delaunay refinement to generate quality meshes for the skin body needs to overcome the following obstacles. First, there is no general piecewise linear representations for smooth surfaces defined by implicit or parametric equations. Although a number of surface polygonization and triangulation algorithms are available, none of them can guarantee the output surface meshes have no acute angles. As a result, the boundary recovery algorithms used in Delaunay refinement for polyhedra are not appliable for meshing the volumes of smooth surfaces. Second, sliver removal algorithms devote further study for the domains bounded by smooth surfaces because Cheng and Dey's algorithm [27] can not handle the boundaries with acute input angles. Third, the mesh size analysis using local feature size of the PLC is not suitable for analyzing the size of volumetric mesh of smooth surfaces. Instead, the curvature and the local feature size of the smooth surface are important for the analysis of the mesh density.

5.3 Algorithm

I present the algorithm to construct quality tetrahedral mesh for the volume bounded by molecular skin surfaces in this section. The algorithm is divided into three stages, namely, *sculpture*, *prioritized Delaunay refinement*, and *sliver removal by pumping vertices*. The procedure of sculpture builds a initial tetrahedral mesh for the skin body. Then, the algorithm improve the mesh quality by running Delaunay refinement and sliver removal sequentially. I firstly introduce the initial tetrahedral mesh builds from a surface mesh constructed by the algorithm in Chapter 4. Then, I describe the prioritized Delaunay refinement methods and analysis the quality guarantees when the refinement terminates. Finally, I adapt the sliver exudation algorithm to generate sliver free quality tetrahedral meshes.

5.3.1 Initial Tetrahedralization of the Skin Body

An initial tetrahedralization of the skin body can be built on the base of the surface triangulation generated by the algorithm in Chapter 4. Recall that the surface triangulation D'_P is the restricted Delaunay triangulation of the ε-sampling P of the skin surface F_B. For each triangle $abc \in D'_P$, it is either a face on the convex hull of P belonged to only one tetrahedron in the Delaunay triangulation D_P of P or a common face shared two tetrahedra in D_P. For the triangle abc that is a face of only one tetrahedron, the circumcenter of the tetrahedron must be inside the skin body.

Otherwise, triangle *abc* is not a restricted Delaunay triangle. If the triangle *abc* is shared by two tetrahedra, one of the tetrahedra must have its circumcenter inside the skin body and the other has its circumcenter outside the skin body so that the line segment passing through the two circumcenters intersect the skin surface. The reason is also that triangle *abc* is a restricted Delaunay triangle and its dual Voronoi is the line segment and it intersects the surface. As a result, the collection of all the tetrahedra whose circumcenter lie inside the skin body forms a initial tetrahedralization T_0 of the skin body. Since the skin surface meshing algorithm built the Delaunay triangulation of P, the initial tetrahedral mesh T_0 can be easily obtained as a side product.

However, there is no any guarantees on the quality of the tetrahedral mesh T_0 although the mesh quality of the surface mesh has lower bound on its minimum angle. Additional vertices inside the skin body are necessary to be inserted for quality improvement. We can apply the Delaunay refinement for this purpose. In particular, the challenges for such an application can be overcomed by utilizing the good surface mesh quality and the nice properties of the restricted Delaunay triangulation of the ε-sampling. Besides the properties described in Lemma 4.2.1 and Lemma 4.2.2, another property of the surface mesh is that the dihedral angle of two neighboring surface triangles has a lower bound, which is presented in the following Lemma.

Lemma 5.3.1 (Dihedral Angle Lemma) *For two triangles $abc, bcd \in D'_P$ with shared edge bc, the dihedral angle at edge bc has a lower bound of $\pi - 2arcsin\frac{2\varepsilon}{1-\varepsilon}$.*

Proof. Consider the two normals of triangle *abc* and *bcd* and their intersection at the point t, see Figure 5.4. The dihedral angle at edge *bc* is equal to $\pi - \angle n_{abc}n_{bcd}$, in which $\angle n_{abc}n_{bcd}$ is the acute angle formed by the two normals n_{abc} and n_{bcd}. Thus, the claim would be true if we can prove the upper bound of the $\angle n_{abc}n_{bcd}$ is $2\arcsin\frac{2\varepsilon}{1-\varepsilon}$.

Denote the normal of the skin surface at the vertex a as n_a. According to the

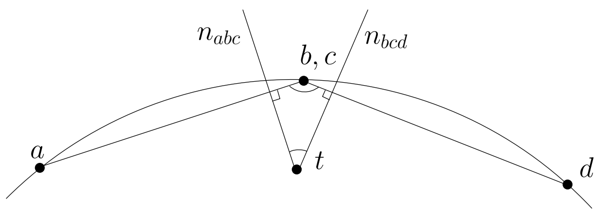


Figure 5.4: The dihedral angle at the edge *bc* of two neighboring surface triangle *abc* and *bcd*.

Triangle Normal Lemma in [23], we know $\angle n_{abc} n_a < \arcsin \frac{2R_{abc}}{\varrho(a)}$. Together with the bound of the circumradius, $R_{abc} < \frac{\varepsilon}{1-\varepsilon} \varrho_{abc}$, which is claimed in the Lemma 4.2.1, we have

$$\angle n_{abc} n_a < \arcsin \frac{2\varepsilon \varrho_{abc}}{(1-\varepsilon)\varrho(a)} < \arcsin \frac{2\varepsilon}{1-\varepsilon}.$$

Similarly, we have $\angle n_{bcd} n_a < \arcsin \frac{2\varepsilon}{1-\varepsilon}$ as well. The angle $\angle n_{abc} n_{bcd}$ is largest when the normal n_a is parallel the plane determined by n_{abc} and n_{bcd}. That is,

$$\angle n_{abc} n_{bcd} \leq \angle n_{abc} n_a + \angle n_{bcd} n_a < 2 \arcsin \frac{2\varepsilon}{1-\varepsilon},$$

which implies the lower bound claimed in the Lemma. $\boxdot$

Lemma 5.3.1 implies that the tetrahedra formed by two neighboring would be a sliver because the minimal dihedral angle of the tetrahedra would approach 0°. At the same time, the circumcenters of such tetrahedra lie close to the surface. For all other tetrahedra in T_0, their circumcenter lie far from the surface but near the media axis of the surface. In order to maintain the surface triangulation stable in the tetrahedral mesh of the skin body, we propose to insert new mesh vertices as far as possible from the surface. In particular, we insert the new mesh vertices with a priority parameterized by the *distance function* to the skin surface.

Distance Function. Given a skin surface F_B, the distance function to F_B is defined over $\mathbb{R}^3$ by assigning each point its distance to the surface, namely,

$$d(x) = \inf_{p \in F_B} \|x - p\|, \forall x \in \mathbb{R}^3.$$

We approximate the function using the ε-sampling P of skin the skin surface, that is,

$$d'(x) = \min_{p \in P} \|x - p\|, \forall x \in \mathbb{R}^3.$$

The approximation has been used by Dey et al.[35] to reconstruct the smooth surface from a point cloud. We use the function value to parameterize the priority for new mesh vertices insertion during the Delaunay refinement. The priority ensures the additional mesh vertices are far from the surface and lie inside the skin body. As a result, the surface mesh are always kept in the tetrahedral mesh of the skin body. For the convenience of the discussion, we call the circumsphere of a surface triangle *abc* the *protecting ball* of the *abc*. The union of the protecting balls of all the surface triangles forms the *protecting region*. The inserting new mesh vertices are guaranteed to be always outside the protecting region. Next, I describe the prioritized Delaunay refinement algorithm to improve the mesh quality of the initial tetrahedralization T_0.

5.3.2 Prioritized Delaunay Refinement

Delaunay refinement methods improve the mesh quality by inserting the circumcenters of the poor shape tetrahedra incrementally. After each new mesh vertex is inserted, the Delaunay triangulation is maintained and this process is repeated until the mesh quality satisfies the constraints. The new mesh vertices can be inserted in a random way or in a certain order. Shewchuk [90] inserted new mesh vertices with a priority parameterized by the radius-edge ratio. That is, the circumcenter of the tetrahedron with largest radius-edge ratio is always inserted. This priority decreases the number of inserting points in some cases. Edelsbrunner and Guoy [42] defined the sink as the circumcenters that are contained inside their own tetrahedra. A circumcenter is inserted as a new mesh vertex only when it is a sink. The priority facilitates parallel implementation of the Delaunay refinement.

We introduce a new priority parameterized by the distance function value of the circumcenter. That is, the circumcenter t of a skinny tetrahedron τ that has the largest distance $d'(t)$ to the surface is inserted in each iteration of the Delaunay refinement. The reason beyond this priority is that new mesh vertices are restricted to be not too close to the surface and form bad shape tetrahedra during the refinement. As a result, the circumcenters close to the media axis of the surface are inserted in high priority to improve the mesh quality as much as possible. When the circumcenters near the surface are necessary to be inserted, the mesh quality satisfies the quality constraints because the input surface mesh has guaranteed quality. See Figure 5.5 for an example in two dimensions. From left to right in the first row, the figures illustrate the process of prioritized Delaunay refinement on the triangular mesh bounded by a skin curve. The minimal angle in the rightmost figure has a lower bound of 30°.

The Delaunay refinement adapts the incremental algorithm for Delaunay triangulation computation. Starting from T_0, each time a circumcenter t_i is inserted and forms four new tetrahedra with the faces of the tetrahedron in T_{i-1} enclosing t_i. The Delaunay property are restored by edge flipping algorithm and we get the Delaunay tetrahedral mesh T_i with mesh vertices $P_i = P_{i-1} \bigcup \{t_i\}$, for $P_0 = P$. The procedure is described in the following pseudo code.

We analyze the behavior of the Algorithm 5.1 to validate the termination and quality guarantees of the prioritized refinement procedure. First, we prove that all the inserted circumcenters lies inside the skin volume V_B and outside the protecting

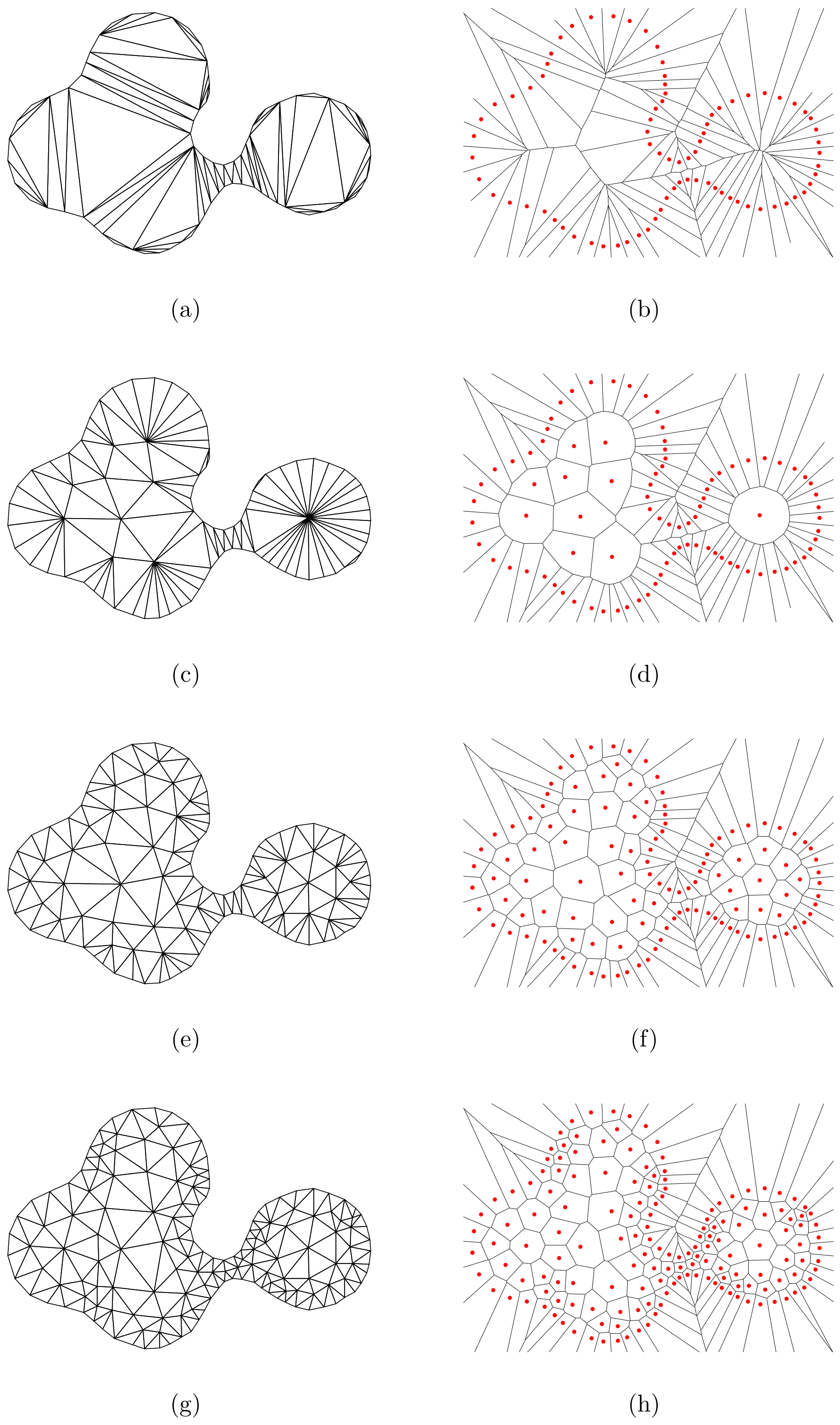


Figure 5.5: Four Delaunay triangulations and the dual Voronoi diagrams restricted by a skin curve generated by the prioritized Delaunay refinement.

Algorithm 5.1 PDeloneRefine()

```
1: Test the radius-edge ratio for all the tetrahedra in $T_0$ and push the skinny tetra-
   hedron to a queue $Q$ prioritized by their distance function value;
2: while $Q \neq \varnothing$ do
3:   $\tau$ = ExtractMax($Q$);
4:   if the $\tau$ is a valid tetrahedron in $T_{i-1}$ with the circumcenter $t_i$ that falls outside
     of the protecting region then
5:     Compute the Delaunay triangulation of $P_i = P_{i-1} \cup \{t_i\}$;
6:     Update the priority $Q$ by adding the new skinny tetrahedra;
7:   else
8:     Continue;
9:   end if
10: end while
```

region. This property ensures the input surface mesh is stable during the refinement process. Then, we prove the refinement process terminates with a upper bound of the radius-edge ratio c depending on the constant ε and γ that specifies the surface mesh quality.

Lemma 5.3.2 (Bounded Circumcenters Lemma) *Let t be the circumcenter of a skinny tetrahedron $\tau \in T_i$. The circumcenter t is contained in the underlying space of T_i, namely, $|T_i|$.*

Proof. We prove the lemma using the deductive method.

In the case of $i = 0$, the claim is true on the base of our sculpture procedure. Since T_0 consists of all the tetrahedra whose circumcenters lies inside the volume of the skin surface V_B. The difference between V_B and $|T_0|$ is the space between the skin surface and the surface triangles when the local shape of the surface is convex. According to the dihedral angle Lemma 5.3.1, a tetrahedra with its circumcenter inside V_B and outside $|T_0|$ must be a sliver. And a sliver is never a skinny tetrahedron. Thus, the claim is true when $i = 0$.

We assume the claim is true when $i = k$ and prove all the circumcenters of tetrahedra in T_{k+1} are inside $|T_{k+1}|$. To get a contradiction, let a circumcenter t

locates outside $|T_{k+1}|$. The circumcenter t must be a Voronoi vertex of a mesh vertex q that either lie on the boundary or the interior of $|T_{k+1}|$. In the case of the mesh vertex q lie inside $|T_{k+1}|$, one of the Voronoi edge of q must penetrate the surface mesh and we denote the intersection with a surface triangle abc as u. As a result, the point u must be inside the protecting sphere of abc and its distance to a is smaller than the distance to q since q is always outside the protecting region. It contradicts with the definition of the Voronoi cell for the point q. In the case of the mesh vertex q is on the boundary, then t must be the circumcenter of a tetrahedron with its four vertices on the boundary. It is impossible because all the new tetrahedra we created during the refinement must connect to the newly inserted internal nodes. Thus, all the circumcenters of tetrahedra in T_{k+1} are inside $|T_{k+1}|$ and the claim follows. ◻

The Lemma 5.3.2 implies that the boundary of the tetrahedral mesh T_n conforms to the input surface mesh. Next, we prove the algorithm terminates with bounded radius-edge ratio for all the tetrahedra in T_n

Thm. 5.3.1 *The algorithm terminates with quality tetrahedra mesh having ratio property for the constant*

$$c \leq \frac{2\varepsilon}{\gamma(1-\varepsilon)}.$$

Proof. We first prove the algorithm terminates. Since the algorithm only inserts the circumcenters of tetrahedra with radius-edge ratio larger than 1, the edge length of newly created edges during the Delaunay refinement never shrinks. In the other words, the inter-vertices distances are bounded from below. Moreover, the algorithm only inserts points in the domain and never deletes any points. As a result, the algorithm must terminate because the volume of V_B is finite.

Then, we prove the bound for the radius edge ratio. To get a contradiction, we assume a tetrahedron τ has radius-edge ratio $\frac{R}{l} > \frac{2\varepsilon}{\gamma(1-\varepsilon)}$ when the algorithm

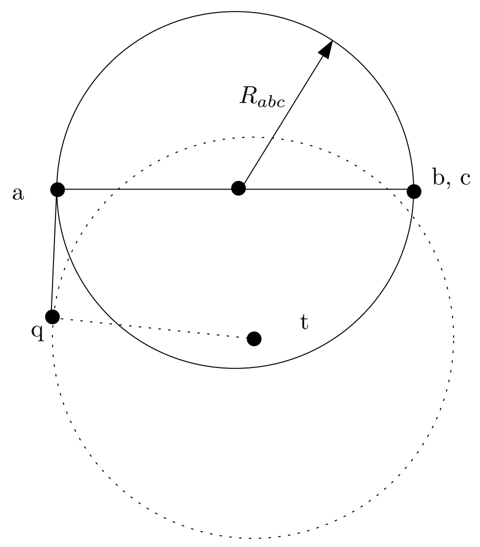


Figure 5.6: A tetrahedron with its circumcenter inside a protecting sphere.

terminates. Then, the circumcenter t of τ must be inside a protecting sphere of a surface triangle abc. See Figure 5.6. Let one of the vertices of τ be q, then we have

$$R = \|t - q\| < \|t - a\| < 2R_{abc},$$

in which R_{abc} is the radius of the protecting sphere of triangle abc. According to Lemma 4.2.1 we get

$$R < \frac{2\varepsilon}{(1-\varepsilon)}\varrho_{abc}. \tag{5.1}$$

On the other hand, the length of the shortest edge of the tetrahedron τ must be longer than the length of any edges on the boundary incident to a. That is,

$$l > \|a - b\| \geq \gamma\varrho_{ab}.$$

According to the assumption,

$$R > l\frac{2\varepsilon}{\gamma(1-\varepsilon)} > \gamma\varrho_{ab}\frac{2\varepsilon}{\gamma(1-\varepsilon)} = \frac{2\varepsilon}{(1-\varepsilon)}\varrho_{ab}. \tag{5.2}$$

Since $\varrho_{abc} \le \varrho_{ab}$, Equation (1) and Equation (2) contradicts each other. As a result, all the tetrahedra has ratio property for the constant $c \le \frac{2\varepsilon}{\gamma(1-\varepsilon)}$. $\Box$

With feasible values for $0 < \gamma < 0.218$ and $0 < \varepsilon < 0.279$, a conservative lower bound for c would be 3.5. Our experiments show that we can achieve a much better bound of 1.5 on the radius-edge ratio.

However, slivers still frequently exist inside the tetrahedral mesh after the Delaunay refinement terminated with a bounded radius-edge ratio. Next, we adapt the sliver exudation algorithm to remove the slivers.

5.3.3 Sliver Removal by Pumping Vertices

Given the Delaunay tetrahedral mesh T_n with ratio property c, the sliver exudation algorithm assigns small weights to the mesh vertices so that the resulting weighted Delaunay triangulation contains no slivers. This procedure can be understood by considering the orthospheres of the tetrahedra incident to the gaining weight vertex in the weighted Delaunay triangulation. See Figure 5.7 for an example in two dimensions. Denote a vertex x with weight w_x^2 as $\hat{x}$ and it can be considered as a sphere centered at x with radius w_x. As the weight of a vertex x increases, the weighted distance between the orthosphere of the triangle xpq and $\hat{x}$ decreases and vanishes when the sphere $\hat{x}$ is orthogonal to the orthosphere of xpq. To maintain the weighted Delaunay triangulation, we need to flip the edge pq to ensure the orthospheres are empty. We call such a weight a *critical weight.* Since a small change of the weight for a vertex can result in dramatic change of the radius of the orthospheres, a feasible weight value for a vertex incident to a sliver would remove the sliver. The weight

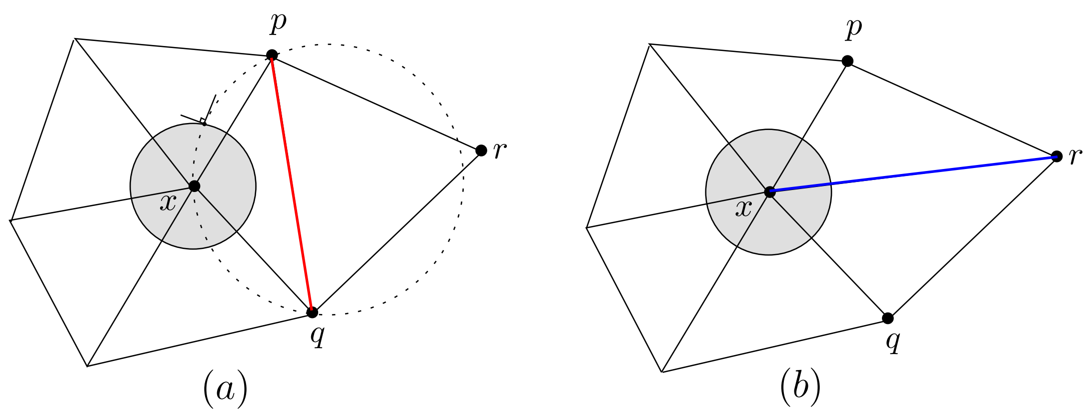


Figure 5.7: Flip an edge to maintain the weighted Delaunay triangulation when the weight of x increases.

assigned w_x^2 to a vertex x is required to be small comparing to the distance to its closest neighbor, that is, $w_x^2 \in [0, \omega_0 N(x)]$, in which $N(x)$ is the distance from x to its nearest mesh vertices and ω_0 is a constant less than one half. This condition ensures that spheres do not intersect each other so that no vertices are deleted in the weighted Delaunay triangulation. Moreover, the small weight assignment will not increase the the radius-edge ratio of weighted Delaunay triangulation very much.

However, the sliver exudation algorithm only applies to the periodic point set, which is an infinite set without boundary. In order to apply the algorithm to our bounded domains, we only pump the non-boundary mesh vertices incident to a sliver. Moreover, we further restrict the assignment of the weights not to challenge the boundary. That is, the weight for x should be small enough so that the weighted distance between $\hat{x}$ with any protecting sphere is positive, which implies that the boundary triangles will stay inside the weighted Delaunay triangulation. For the slivers with four mesh vertices on the surface, we remove them from the tetrahedral mesh directly. In our implementation, we use a threshold for $\varsigma = 5°$ and define a tetrahedron as a sliver if its minimal dihedral angle smaller than ς and the radius-edge ratio is smaller than 1.5. The following pseudo code illustrates the procedure of the sliver removal.

Algorithm 5.2 SliverRemoval()

```
1: for all slivers $\tau$ in $T_n$ do
2:   if $\tau$'s vertices set of $\{p, q, r, s\} \subset P$ then
3:     Remove the sliver $\tau$ from $T_n$;
4:   else
5:     for a mesh vertex $t \in \{p, q, r, s\}$ and $\{t\} \not\subseteq P$ do
6:       $w_\tau^2 =$ pumpVertex($t$);
7:       Set the weight $w_\tau^2$ to $t$ and maintian the weighted Deluanay triangulation;
8:     end for
9:   end if
10: end for
```

The Algorithm 5.2 firstly checks if the four mesh vertices of a sliver are on the boundary F_B. If so, the sliver must have two neighboring surface triangular faces. The reason is that the circumsphere of a sliver with four mesh vertices in other configurations would not be empty, which contradicts with the Delaunay property. According to Lemma 5.3.1, we can simply remove the sliver without influencing the accuracy of the boundary approximation.

For a sliver with at least one boundary mesh vertex, we use the procedure pumpVertex(t) to find the optimal weight for a non boundary vertex so that dihedral angles are maximized locally. In particular, we look through all the critical weights in the interval $w_t \in [0, \omega_0 N(t)]$ that stimulate an edge flipping. We perform each flip and compute the minimal dihedral angle of the tetrahedra incident to t after the flip. If the minimal dihedral angle increases, the critical weight is recorded. When there is no more feasible flips can be proceeded, we return the critical weight that maximize the dihedral angle and restore all the flips performed after this critical weight. The pseudo code of the procedure pumpVertex(t) is illustrated as following:

The Algorithm 5.3 uses a priority queue W to store the critical weights so that the edge flips are performed in an order specified by the increasing weights. Each edge flip is either a two to three flip that replace two tetrahedra sharing a link triangle

Algorithm 5.3 pumpVertex(t)

```
1: Find all the triangles in the link of $t$ that do not belong to $R_B(P)$ and put them
   to a list $L$;
2: For each triangle $pqr \in L$, compute the weight $w_i^2$ for $t$ so that the weighted
   distance between $\hat{t_i}$ and the orthosphere of triangle $pqr$ is 0; push $w_i$ to a priority
   queue $W$ if $w_i \leq \omega_0 N(t)$;
3: Compute the minimal dihedral angle incident to $t$ and set it to $\varsigma_\tau$; set the initial
   value for optimal wight$w_\tau = 0$;
4: while $W \neq \emptyset$ do
5:    $w$ = ExtractMin($W$);
6:    Perform an edge flip $f(w)$ to maintain the weighted Delaunay triangulation;
7:    Compute the minimal dihedral angle $\varsigma$ for all the tetrahedra in the star of $t$;
8:    if $\varsigma > \varsigma_\tau$ then
9:       $w_\tau = w$
10:      $\varsigma_\tau = \varsigma$;
11:   end if
12:   Update the priority queue $W$ by computing new critical weight using the up-
      dated link triangles of $t$;
13:   Push the edge flip $f(w)$ to a stack $F$ and perform a revers flip correspondingly;
14: end while
15: Pop all the flips in $F$ before $f(w_\tau)$;
16: Return $w_\tau$;
```

of t with three tetrahedra or vice visa. If an edge flip increases the dihedral angle, the optimal weight and the minimal dihedral angle are updated. After each flip, the link of t is updated and we push the new critical weights to the priority queue W. All feasible flips are performed when the W is empty. We perform reverse flips to restore the flips that do not increase the dihedral angle. That is, we replace three tetrahedra with two if a two to three flips were performed or vice visa. This operation is facilitated by a stack to store all the performed flips.

5.4 Results

We implemented the algorithm on the PC platform with C++ on the base of the skin surface meshing software built by the authors. The construction of the Delaunay triangulation and weighted Delaunay triangulation partially reuse the prior software on alpha shapes. One point worth noting here is the computation of the distance function value for each circumcenter of a skinny tetrahedra. We utilize the Delaunay triangulation of the input surface mesh vertices for this purpose. That is, we locate the tetrahedron contains the circumcenter first and search its nearest neighbor locally. We test our implementation to generate quality tetrahedral meshes for a few molecular skin models. The experimental results show that the prioritized Delaunay refinement performs excellently and it achieves an upper bound of 1.5 on the radius-edge ratio. At the same time, the dual Voronoi diagram the Delaunay triangulation decompose the volumes into well shaped convex polyhedra. See the second row of the Figure 5.5. Such a decomposition may be useful for the numerical computations using control volume methods. Moreover, our implementation of the sliver exudation algorithm eliminates most of the slivers. Since the dense tetrahedral meshes are hard to be visualized with two dimensional figures, we collect the statistic of the mesh quality on two experiments.

Table 5.1 illustrates the statistics of the mesh quality for the molecule Crambin. The input surface mesh includes 27,341 mesh vertices and 50,222 triangular faces. The minimum angle in the surface mesh is 20.1°. The Delaunay refinement takes around 8 minutes on Pentium 4 PC to insert 26,709 vertices inside the volume and improve the radius-edge ratio to 1.5. Totally 1,296 slivers exist in the final tetrahedral mesh, in which 300 slivers have four vertices on the surface. After we performed the sliver removal, only 13 slivers are left. The distribution of the radius-edge ratio and minimal dihedral angle in the the coarse tetrahedral mesh before prioritized Delaunay refinement T_0, the mesh after the Delaunay refinement T_n and the final mesh after the sliver removal $\hat{T_n}$ is presented in the table 5.1.

Table 5.2 illustrates the statistics of the mesh quality for the molecule PdB7. The display conventions are the same as in Table 5.1.

r/l	0-1	1-1.5	1.5-2	2-3	≥ 3
T_0	2,675	4,451	3,327	25,570	47,726
T_n	117,854	138,609	2	0	0
$\hat{T_n}$	117,654	13,8258	12	0	0
$\varsigma(°)$	0-5	5-10	10-20	20-30	≥ 30
T_0	6,964	12,254	10,787	7,335	46,409
T_n	1,292	3,309	14,009	35,877	201,978
$\hat{T_n}$	13	212	14,479	39,566	201,654

Table 5.1: The distribution of the radius-edge ratio and minimal dihedral angles of the tetrahedral mesh of the molecule Crambin.

r/l	0-1	1-1.5	1.5-2	2-3	≥ 3
T_0	244	405	366	8,131	14,025
T_n	20,296	27,061	1	0	0
$\hat{T_n}$	20,328	27,127	3	0	0
$\varsigma(°)$	0-5	5-10	10-20	20-30	≥ 30
T_0	1,801	3,914	3,314	1,676	12,466
T_n	245	702	2,627	6,402	37,382
$\hat{T_n}$	6	82	3,046	6,547	37,684

Table 5.2: The distribution of the radius-edge ratio and minimal dihedral angles of the tetrahedral mesh of the molecule Pdb7.

5.5 Summary

In this chapter, I present an algorithm for generating quality tetrahedral mesh of the skin body. The algorithm improves the mesh quality of a coarse mesh using Delaunay refinement prioritized by the distance function value followed by a sliver removal process. The prioritized Delaunay refinement process terminates with guarantees on the upper bound of the radius-edge ratio of the tetrahedral mesh. The experimental results shows that an upper bound of 1.5 on the radius-edge ratio can be achieved, which is much better than the theoretical bound. The sliver removal process removes the slivers effectively and improves the minimal dihedral angle in the mesh. The boundary of the final tetrahedral mesh also approximates the original surface accurately.

We observe that the tetrahedral meshes generated by our prioritized Delaunay refinement are well graded from the experimental results. It would be interesting to further analyze the gradedness and the complexity of the mesh using the distance function and the local length scale of the surface. We are also interested in extending our algorithm to generate quality tetrahedral meshes for the domain bounded by other smooth surfaces. Since the input surface mesh may not have guaranteed quality and may not be a subset of the Delaunay triangulation of the surface mesh vertices either, we propose to insert points on the surface during the refinement process. Surface mesh is updated locally and the mesh quality for the surface and volume are improved simultaneously. A critical issue in this extension is to prevent infinitely many surface points are added.

Chapter 6

Skin Meshing Software and Applications

The skin meshing software is implemented with graphical user interface (GUI) on the Windows platform for the construction of skin surface triangulation, Delaunay tetrahedralization of the skin body, alpha shapes and its pockets. The software is the implementation of the meshing algorithms discussed in the previous chapters. The skin meshing software is efficient, robust and user friendly [3]. Since the skin surface lends itself as a desirable surface model of the molecules, the software serves as a powerful computational tool for the study of the molecular shape and function.

In this chapter, I first introduce the software functionalities and demonstrate some results generated by the software. Then, I describe the potential applications of the software.

6.1 Skin Meshing Software

The skin meshing software is developed on the Windows platform with C++ language and OpenGL as the graphics library [17]. The input is the coordinates of the centers and the radii of a set of spheres. In addition, the software can import the PDB files downloaded from the Protein Data Bank [2] and extract the positions and van der Waals radii of the atoms in the molecule specified by the PDB file as input. Users can obtain the molecular skin model of the molecule by specifying parameters to set the radius as $\sqrt{2}$ times the summation of the atom's van der Waals radius and the radius of the probe sphere, which is usually chosen as 1.4 Angstrom to represent the water as solvent. The user interface of the software is illustrated in Figure 6.1.

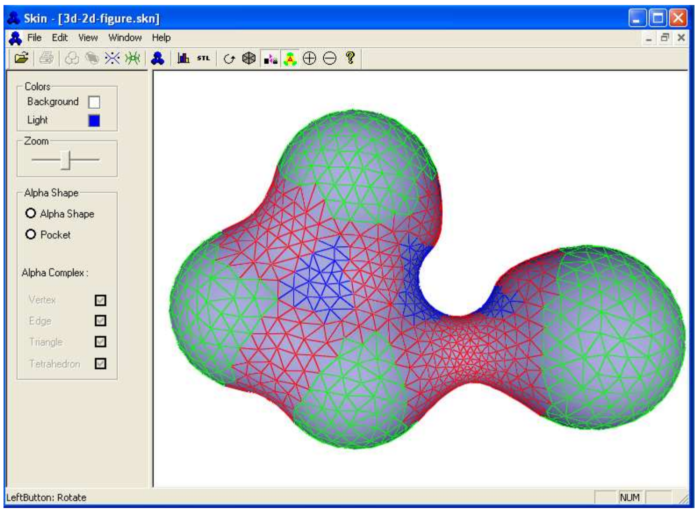


Figure 6.1: The user interface of the skin meshing software.

The software outputs guaranteed quality meshes for both the surface and the volume of the skin surface specified by a set of spheres or the molecular skin model of the input PDB file. Moreover, the software can compute the alpha shapes and the pockets as well [41]. In the following, I describe the capabilities of the skin meshing software using a few examples.

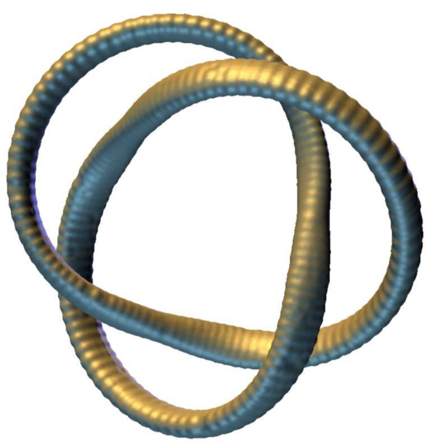
Surface Triangulation. Figure 6.2 shows a few examples of the skin surface generated by the skin meshing software. Figures 6.2 (a) and (b) illustrate two tubes modeled by the skin surface. Each tube is defined by a set of spheres centered on the a curve defined by a equation. Figure 6.2 (c) and (d) show the skin model of a cactus and a human hand respectively. These two models are defined by the inner poles, which are set of spheres computed by the PowerCrust [7] using a set of sample points one the surface. Figure 6.2 (e) illustrates the molecular skin model of a protein with PID 1ach. The zoomed in view of the partial mesh in the rectangle in Figure 6.2 (e) is showed in Figure 6.2 (f).

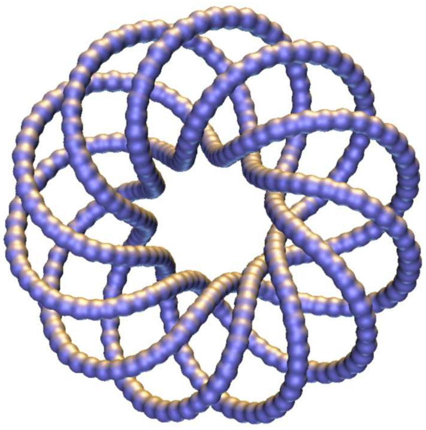
These results demonstrate that our meshing algorithms can generate homeomorphic surface triangulations for skin surfaces with arbitrary topology, for example, the complicated nested knots shown in Figure 6.2(b). The reason is that the surface triangulations generated by our algorithm are the restricted Delaunay triangulations of ε-samplings of the skin surface, which have the closed ball property that guarantees the homeomorphism. Moreover, the surface triangulations approximate the surface geometry accurately. The topological features such as genus and tunnels in the molecular models are preserved in the meshing results as well, as shown in Figure 6.2(f). This result contrasts with the previous results in [9] that small genuses in a molecule were usually wrongly triangulated.

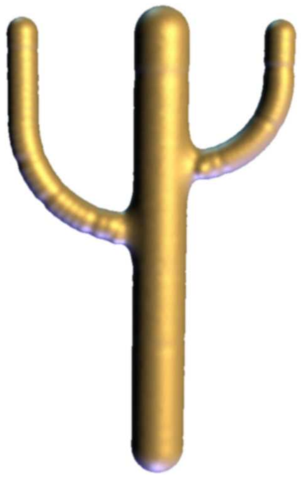
molecular name	no. triangles in the mesh	minimum angle distribution(%)			
		50°-60°	30°-50°	20°-30°	Less than 20°
$Pdb7(Figure3.7(b))$	24,336	59.46	40.30	0.24	0
$HIV2(Figure4.1)$	226,758	56.22	43.54	0.24	0
$1CHO(Figure1.4)$	253,024	56.00	43.77	0.22	0.01
$1ACB(Figure6.2(e))$	290,476	56.20	43.56	0.2397	0.0003

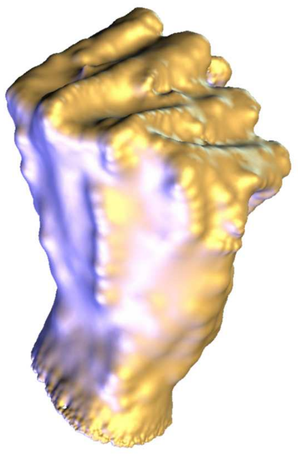
Table 6.1: The statistics of the minimum angle of the triangles in the surface mesh.

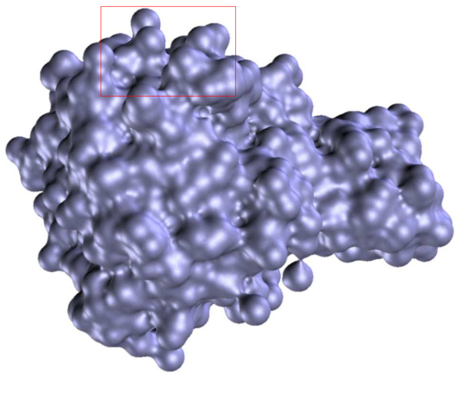
Table 6.1 lists the statistical results of the mesh quality of a few molecules illustrated in this thesis. The results indicate that our implementation achieved a better bound on the minimum angle of the triangulation than the theoretical result. For example, the minimum angle in the triangulation of pdb7 molecule is 27.02°, which is much better than the theoretical result 21°. Moreover, most triangles in the mesh are almost equilateral triangles, that is, more than 50% triangles in the mesh have


(a)


(b)


(c)


(d)


(e)

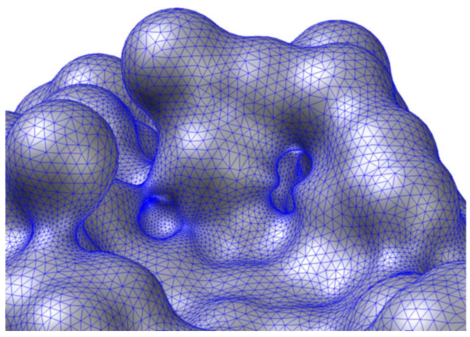

(f)

Figure 6.2: Examples of the skin surface generated by the skin meshing software.

minimum angles around 60°. Only very few triangles have minimum angle less than 30°. The high quality we achieved may be explained by two main factors: first, the smoothness of the molecular skin model is crucial to generate quality meshes. It would be problematic to guarantee a lower bound on the minimum angle of the triangulation if the surface is not smooth. Second, our strategy of placing new mesh vertices adaptive to the surface curvature creates well-shaped triangles.

molecular name	no. triangles		minimum angle		computing time		
	Alg. 4	Alg. 3	Alg. 4	Alg. 3	Alg. 4	Alg. 3	dynamic
$Pdb7$ (Fig. 3.7 (b))	19,000	24,336	23.16°	27.02°	00:00:14	00:00:05	00:10:05
$A-DNA$ (Fig. 3.15)	104,961	114,316	20.92°	24.12°	00:02:14	00:00:51	00:35:12
$GramicidinA$ (Fig. 3.16)	189,188	305,186	19.09°	24.37°	00:04:46	00:03:33	01:35:23
$HIV2$ (Fig. 4.1)	226,758	N.A.	20.35°	N.A.	00:05:36	N.A.	N.A.
$1CHO$ (Fig. 1.4)	253,024	N.A.	19.16°	N.A.	00:08:03	N.A.	N.A.
$1ACB$ (Fig. 6.2 (e))	290,476	N.A.	19.53°	N.A.	00:10:08	N.A.	N.A.

Table 6.2: Comparison of the performance between the two surface meshing algorithms in Chapter 3 and Chapter 4.

Table 6.2 lists the statistical results of several surface meshes demonstrated in the thesis, along with a comparison of the computation time between the algorithms described in this thesis and the dynamic skin triangulation algorithm implemented by Cheng [23]. All these experiments were run on a Pentium 4 PC. The results clearly demonstrate that our algorithms achieved high efficiency. For example, the computing time of the molecule Gramicidin A decreased from more than 1 hour to 3 minutes. The dramatic improvement of the efficiency can be explained by using the advancing front method to avoid the costly computation of 3D Delaunay triangulation. Moreover, the usage of the Morse-Smale complex for solving front collision problems plays an important part in preventing overlapping triangles efficiently. Conventional methods to handle this problem are always time consuming because the potential collisions are detected by frequently checking the distance of any two vertices on the front [60]. Comparing with the skin meshing algorithm using restricted union of balls, the adaptive sweeping skin meshing algorithm achieves an better mesh quality but generates more triangles, as shown in the Table 6.2. On the other hand, the skin meshing algorithm using restricted union of balls performs more robustly. That is, the skin meshing algorithm using restricted union of balls can generate triangulations of a few molecular models that fail the adaptive sweeping skin meshing algorithm. The main reason for the failure is that the bumpy appearance of the surface results in a large number of noisy critical points. During the simplification of these noisy critical points, some critical points are removed wrongly because of the accumulation of numerical errors in the approximation of the Morse-smale complex. The skin meshing algorithm using restricted union of balls can handle the bumpy surface robustly because it does not use the Morse-smale complex. Moreover, the robust and stable implementation of the alpha shape software relieve us from the robustness worries associated with the Delaunay triangulation computation.

Tetrahedralization of the Skin Body. The skin meshing software can generate quality tetrahedral meshes for the volumes bounded by skin surfaces using the algorithm described in Chapter 5.

Figure 6.3 (a) shows an input surface mesh that is a triangulation of a sphere. The cross section of the surface mesh and the initial tetrahedralization with a plane are shown in Figure 6.3 (b) and (c) respectively. After the prioritized Delaunay refinement is applied to the initial mesh, we obtain a tetrahedral mesh with bounded radius-edge ratio, as shown in 6.3 (d). Five slivers still exist in the tetrahedral mesh and they are eliminated by bumping the mesh vertices. Figure 6.3 (e) illustrates the left slivers contained in the transparent boundary surface. Figure 6.3 (f) illustrates

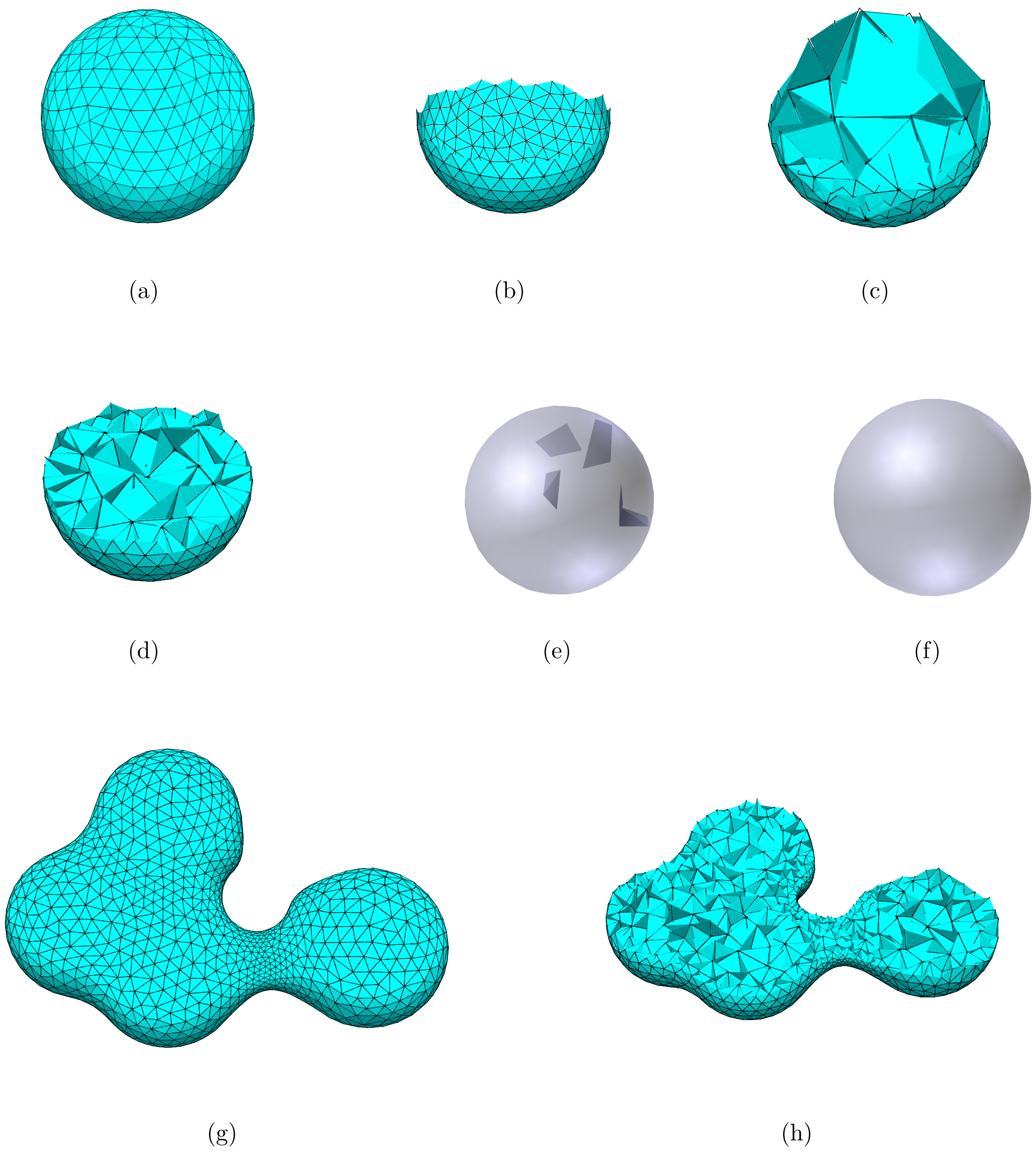


Figure 6.3: Experiments result of the quality tetrahedral mesh generation.

the result after the sliver removal. Another example of the tetrahedral meshes is shown in Figure 6.3 (g). Its cross-section with a plane is shown in Figure 6.3 (h).

These results indicate that the tetrahedral meshes conform to the boundary surfaces meshes. This can be explained by two factors. First, the input surface meshes are the subset of the Delaunay triangulation of an ε-sampling of the skin surface so that we can build an initial tetrahedral mesh conforms to the boundary. Second, the prioritized Delaunay refinement inserts new mesh vertices as far as possible from the surface. Together with the good surface mesh quality, the tetrahedra next to the boundary can satisfy the quality constraints without breaking the boundary. This boundary conformness property is desirable when solving the Poisson-Boltzmann equation with the finite element methods since the solution is sensitive to the boundary. Moreover, the resulting tetrahedral meshes are well graded with good quality, as shown in the Figure 6.3 (d) and (h). The experimental results also show that our algorithm achieves much better bound on the radius-edge ratio of the tetrahedral mesh in practice than the theoretical bound. For example, the lower bound of the radius-edge ratio for the tetrahedral mesh in Figure 6.3 (h) is 1.2. The good quality mesh can facilitate fast and accurate numerical methods for solving the partial differential equations.

Cavities and Pockets of Molecular Skin Models. The skin meshing software can also identify the geometrical features such as cavities and pockets of a molecular skin model F_B. A cavity of the skin F_B is a void contained inside the skin surface, which is a bounded component in the complement space of the skin body, namely, $\mathbb{R}^3 - \text{body}(B)$. A pocket of the skin F_B refers to a concavity feature on the surface. In particular, these concavities open up to the outside with narrow entrances. We define the pockets of a skin surface as the intersections between the skin surface F_B and the pockets in the alpha shape K_B. A pocket in the alpha shape K_B is a portion of the complementary space $\mathbb{R}^3 - \bigcup B$ that have some narrow entrances from the outside [41].

Figure 6.4 (a) shows the van der Walls model of a protein molecule with PID 1scn. The alpha shape specified by the molecule is shown in Figure 6.4 (b). The cavities and pockets of the alpha shape are illustrated in Figure 6.4 (c). Either a cavity or a pocket is a collection of the tetrahedra that do not belong to the alpha shapes but are contained in the weighted Delaunay triangulation. A cavity differs from a pocket in that the pocket has opens to the outside of the alpha shapes. Figure 6.4 (d) shows the molecular skin model of the protein 1scn. The cross-section of the molecular skin model is illustrated in Figure 6.4 (e) and the red color regions identify the cavities inside the molecule. Figure 6.4 (f) illustrates the largest pocket on the surface of the protein 1scn.

These results demonstrate that the skin meshing software generates desirable geometric representations for the molecules. First, the alpha shapes can be used to compute geometric measures of a molecule, such as its volume and surface area [74]. Second, the pockets of the alpha shapes are elegant mathematical models for the binding sites of the proteins. The definition of the pockets do not rely on the size of the probe sphere and grid resolution so that they are free of the numerical issues in the work like [68]. Third, the molecular skin model represents the shape of the molecule better than the exiting surface models of molecules. On one hand, it represents the geometry of the molecules more accurately than the van der Walls model because the surface of the molecular skin model is smooth while the other is not. Moreover, the cavities inside the molecules can be identified and represented only in the skin model. These results also certify that the molecular skin model is a better geometric model for the macro-molecules such as proteins and DNAs. Finally, the pockets of the molecular skin model capture the concave geometric features on the surface of a protein. These features are the key to study the function of the proteins.

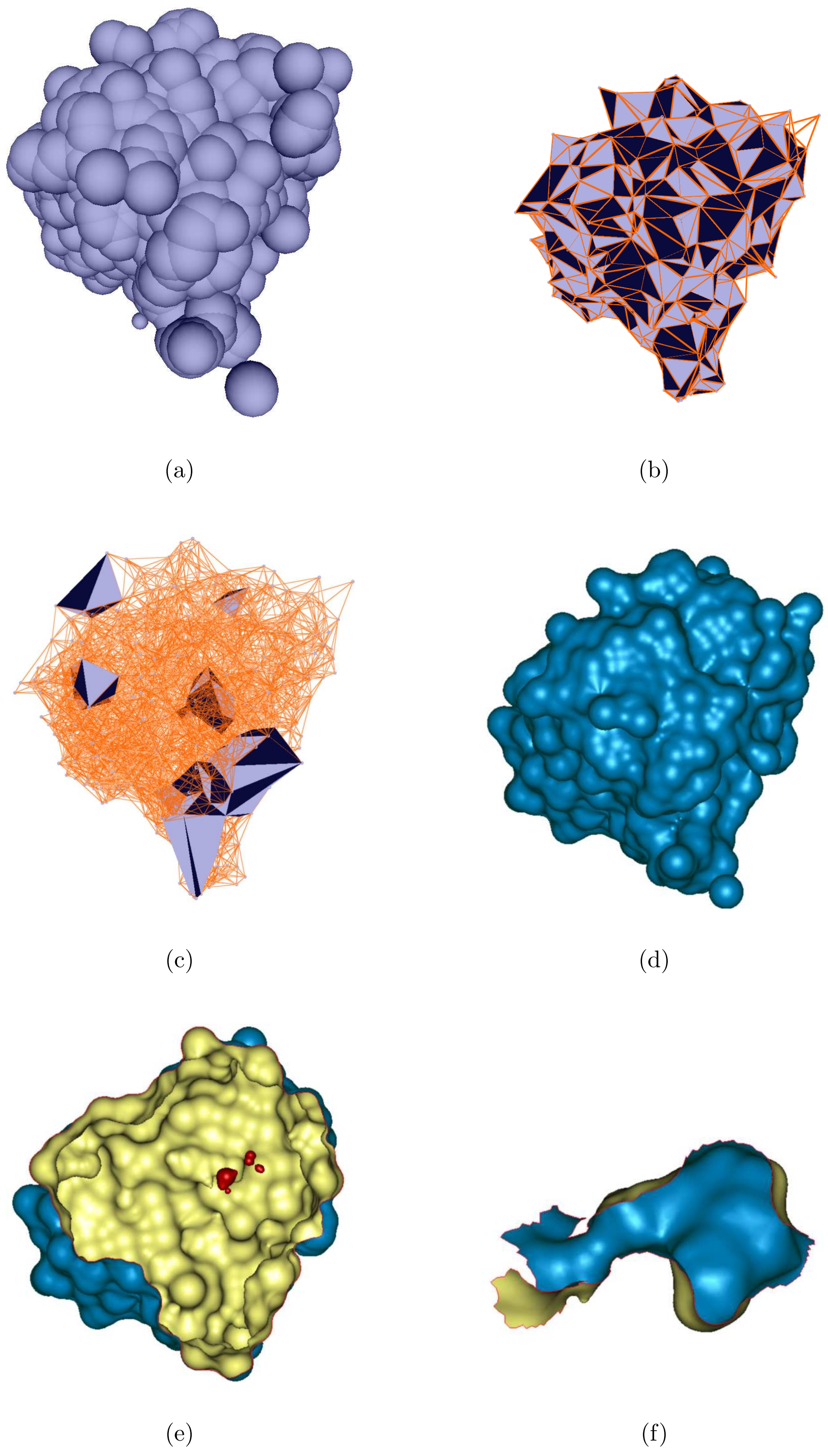


Figure 6.4: Molecular models generated by the skin meshing software.

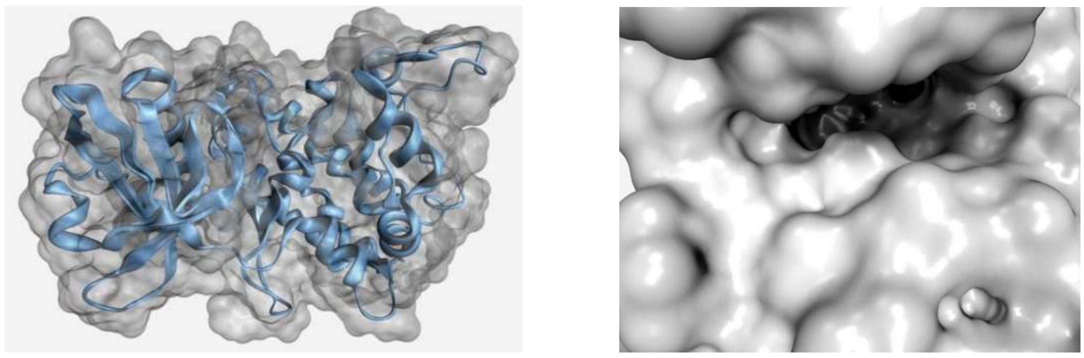
In summary, the skin meshing software can be a powerful computational tool for the study of geometric shapes and the functionalities of molecules. Next, I propose a few potential applications of the skin meshing software.

6.2 Applications

The skin meshes generated by the skin meshing software have applications in computational biology, geometric modeling and computer graphics. I will describe four closely related applications, namely, molecular visualization, molecular shape analysis, numerical computations, and computer graphics.

Molecular Visualization. Graphical models of a molecule provide valuable information about the structure, physical and chemical properties, and biological functions of the molecule [93]. The skin surface lends itself as a surface model for the macromolecules with desirable properties like smoothness and complementarity, which are not shared by other models. The rendering of the skin surface meshes visualizes the molecular shape with a smooth surface, as shown in Figure 6.5 (a). It complements the current molecular visualization styles using ball-stick model, cartoon model and etc. Moreover, the shape features like concavities can be identified on the skin model. See 6.5 (b) for an example.

In addition, the skin meshes also facilitate the visualization of the molecular properties, such as atomic charge, electrostatic potential, polarization etc. We can encode the information as color codes and texture maps over the skin surface to represent these added dimensional properties. For example, the red colored regions on the surface represent the areas with negative polarization and the blue colored regions have positive polarization.



(a) (b)

Figure 6.5: Molecular models of a protein. Figure (a) shows molecular skin model with the cartoon model embedded inside and (b) illustrates the zoomed view of a concavity on the surface.

Molecular Shape Analysis. The molecular shape plays an important role in the process of the protein-ligand interactions. Reactions always occur when the shapes of two proteins fit each other. To predict whether two proteins A and B would interact each other, we can develop algorithms to find the best transformation of B such that it best complements the protein A. That is, the surfaces of the protein A and B match each other partially without intersections. A main issue involved in here is to improve the efficiency of the searching algorithm because it is computationally infeasible to rotate and translate one surface model for exploring every possible transformation to match the other. We can match the essential geometric features on the surface instead of matching the whole shape globally. For example, Agarwal et al. [4] uses the critical points of a scalar function defined over the molecular surface to characterize the concavities and protrusions. The surface triangulations generated by the skin meshing software facilitates efficient combinatorial algorithm to compute such geometric features. As a result, the application of our skin meshes to the protein-ligand docking problem can achieve high efficiency.

Moreover, the skin meshes are useful in the applications of finding the binding sites of proteins. A binding site of a protein is the region on the protein surface where a ligand interacts with the protein, which are important to predict the protein-ligand interactions [86]. According to the experimental results, the binding sites usually located in the concavities such as the clefts and grooves on the surface of the proteins [68]. As a result, the pockets on the skin surface would be positive conjectures for the binding sites. There are always more than one pockets on a skin surface and we need to filter the false positive ones. We can compute shape descriptors [8, 76, 88] of the triangular patches representing the pockets. Then, we apply machine learning methods on the existing true binding site database to derive rules that describe the relationship between the true binding sites and the pockets of the skin surface. These rules would suggest new methodologies to study the protein-ligand problem.

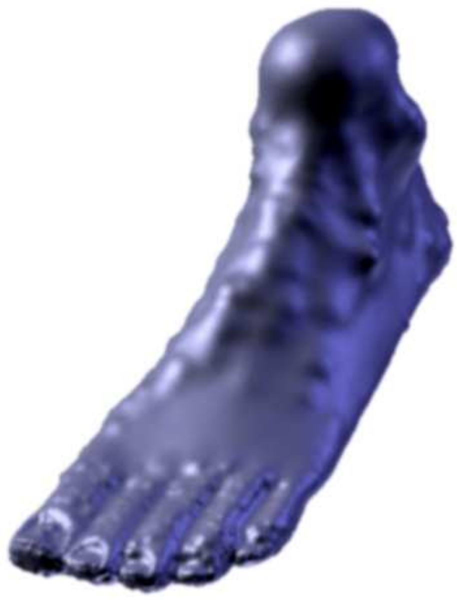
Numerical Computations. An important feature of the meshes generated by the skin meshing software is that both the surface triangulation and the tetrahedralization of the skin body have guaranteed quality. This property is desirable for the numerical computations over the surface and the volume bounded by the surface because the accuracy and stability of the solution with numerical methods depend on the mesh quality. For example, the computation of the electrostatic potential of molecules by solving the Poisson-Boltzmann equation(PBE).

The quality surface meshes facilitate the boundary element methods (BEM) to solve the linear partial differential equations defined over the domain bounded by the skin surface. The BEM methods exploit the fact that many differential equations can be transformed into a set of integral equations over the boundary surface. With a discrete approximation of the boundary, namely, the surface triangulation, the solution of the integral equations can be approximated by the summation of the integration over each triangle. This can be carried out by numerical quadrature, in which the function value at each point inside the triangle is the linear interpolation of the known value at the mesh vertices of the triangle. The interpolation error would be minimized when the minimal angle of the triangle is maximized. Since the surface triangulation generated by the skin meshing software guarantees a lower bound on the minimum angle and approximates the surface topologically correct and geometrically accurate, I believe the solution of the BEM using the skin triangulation will be more accurate.

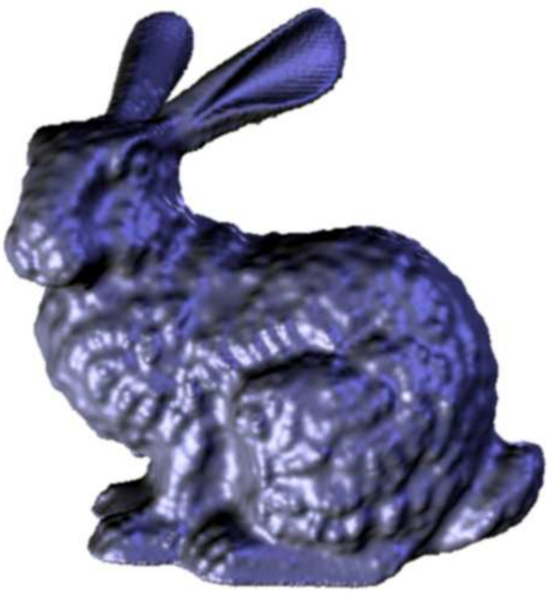
The tetrahedral meshes of the skin body can be applied in the finite element methods for solving the nonlinear PBE. There are three advantages of our tetrahedral meshes over the structure grids used in the previous work [63, 84]. First, the tetrahedral mesh is adaptive and well graded. In other words, the region inside the domain is filled by large tetrahedra and the region near the boundary consists of small tetrahedra. As a result, we have a smaller number of tetrahedra than that of the grids to decomposing the domain, which implies a faster solution for the FEM methods. Second, the boundary of tetrahedral mesh approximates the original surface accurately. This property is important for accurate solution because the PBE is sensitive to the boundary. Contrast with the numerical errors due to the limited grid resolution in [84], our tetrahedral meshes conform to the boundary perfectly. Finally, the tetrahedral mesh has guarantees on the radius-edge ratio and is free of slivers. These properties are essential for the accuracy and convergency of the finite element methods. As a result, the solution of PBE using the finite element methods over our tetrahedral mesh will achieve good agreements with experimental results.

Computer Graphics. The skin surface will be a supplementary representation for the geometric objects with smooth surfaces. Currently, there are mainly two groups of smooth surfaces: explicit and implicit surfaces [12]. The most popular explicit surfaces are parametric surfaces, for example, nonuniform rational B-splines(NURBS), which are widely used in the computer graphics and computer aided design applications [55, 85]. The advantages of the parametric surfaces include ease of triangulating and multiresolution control. However, modeling the details on a surface needs a large number of parametric surface patches. Other than parametric surfaces, implicit surfaces can model high degree of smoothness for arbitrary surfaces. They are capable of deformation and ray tracing but can not be easily triangulated and parameterized [16, 82].

The skin surface captures the advantages of both parametric surfaces and implicit surfaces. On one hand, the skin surface is a piecewise quadratic surface that is capable of being parameterized and triangulated. On the other hand, the skin surface can be deformed freely with smooth transitions. Cheng et al. [24] had established a new framework of automatic shape deformation based on the skin representation. Therefore, the skin surface has the potential to be used widely in the applications such as geometric design and computer animation [55, 82, 100]. Figure 6.6 illustrates a foot and the Stanford Bunny represented by the skin surface.



(a)



(b)

Figure 6.6: The skin model of a foot and the Stanford Bunny.

Chapter 7

Conclusion

This thesis developed efficient algorithms to generate high quality surface and volumetric meshes for macromolecules. The meshes will improve the accuracy of protein-ligand docking significantly. On one hand, the surface meshes facilitate efficient algorithms to compute the alignments of two molecules with perfect shape matching. On the other hand, the volumetric meshes are necessary to compute the electrostatic potential of macromolecules with numerical methods. The electrostatic potential helps to discriminate the real docking conformation from the set of alignments. As such, this work is expected to advance the research in the field of drug development because accurate protein-ligand docking improves the efficiency of drug selections and decreases the costs of experiments for drug tests simultaneously.

Besides the contribution to molecular modeling, this thesis also contributes to the research in mesh generation in three aspects. First, the front collision problem in advancing front methods is handled by employing the critical points of a Morse function. Our experimental results show that the solution improves the efficiency of advancing front meshing algorithms dramatically. Second, a new meshing algorithm is developed by integrating the Delaunay triangulation into advancing front methods. The algorithm captures the advantages of both front advancing and Delauany-based meshing methods. On one hand, the algorithm places mesh vertices incrementally with precise control and performs efficiently. On the other hand, the algorithm provides provable guarantees on the mesh quality. Third, a variant of the Delaunay refinement, namely, the prioritized Delaunay refinement, is applied to generate quality tetrahedral meshes for the volumes of smooth surfaces. The priority of new mesh vertices insertion is parameterized by the distance function defined by the surface. Such a priority enables the Delaunay refinement to generate well-graded tetrahedral meshes that conform to their boundary. The well-graded and conformal tetrahedral meshes improve the accuracy of the solution of numerical methods and accelerate the convergency of the solvers [20].

Further investigation is expected in several aspects. First, we can extend our sweeping meshing algorithm to triangulate the implicit surfaces. In such an extension, only one issue needs to be addressed, that is, the reformulation of the curvature adaptive schema. One choice is to use the estimation of the local feature size to formulate triangle size control constraints because its variation satisfies the one-Lipsitz condition. Next, we can apply the sweeping algorithm to mesh a parametric surface patch such as Nonuniform Rational B-Splines (NURBS) as well because quality meshes for NURBS are desirable in current computer aided design studies [85]. The boundaries of the surface patch can be first split to a collection of edges. These edges work as the initial front and facilitate the application of our sweeping algorithm because the surface patch is not a closed surface anymore and we cannot use the minimum points. I believe these extensions would achieve similar meshing results to our current results. Finally, the application of the skin surface is not limited to molecular modeling. It is possible to investigate the application of the skin surfaces in the computer graphics. Kruithof et al. [69] stepped toward this direction by approximating a simple smooth surface by a skin surface. Cheng and Tan [26] also proposed a method to approximate polygonal objects with skin surfaces. Because the skin surface can be deformed freely with smooth transitions, the approximation of a surface model with skin surfaces will give new insights for the computer animation studies [82, 100].

Bibliography

[1] Alpha shapes: http://biogeometry.duke.edu.

[2] Protein data bank. In *http://www.rcsb.org*.

[3] Skin meshing software. In *http://www.comp.nus.edu.sg/ shixinwe/software.htm*.

[4] Agarwal, P. K., Edelsbrunner, H., Harer, J., and Wang, Y. Extreme elevation on a 2-manifold. In *Proceedings of the 20th annual symposium on Computational geometry* (2004), pp. 357 – 365.

[5] Akkiraju, N., and Edelsbrunner, H. Triangulating the surface of a molecule 71 (1996), 5-22. In *Discrete Appl. Math.* (1996), vol. 71, pp. 5–22.

[6] Akkouche, S., and Galin, E. Adaptive implicit surface polygonization using marching triangles. In *Computer Graphic Forum* (2001), vol. 20, pp. 67–80.

[7] Amenta, N., Choi, S., and Kolluri, R. K. The power crust, union of balls, and the medial axis transform. In *International Journal of Computational Geometry and its Applications* (2001), vol. 19, pp. 127–153.

[8] Ankerst, M., Kastenmller, G., Kriegel, H.-P., and Seidl, T. 3d shape histograms for similarity search and classification in spatial databases. In *Proceedings of the 6th International Symposium on Advances in Spatial Databases* (1999), Springer-Verlag, pp. 207–226.

[9] Bajaj, C., Lee, H. Y., Merkert, R., and Pascucci, V. Nurbs based b-rep models for macromolecules and their properties. In *Proceedings of the fourth ACM symposium on Solid modeling and applications* (1997), pp. 217 – 228.

[10] Baker, N. A., and McCammon, J. A. *Structural Bioinformatics.* Wiley-Liss, Inc., 2003, ch. Electrostatic Interactions, pp. 427–440.

[11] Bern, M., Eppstein, D., and Gilbert, J. Provably good mesh generation. In *J. Comput. Syst. Sci.* (Orlando, FL, USA, 1994), vol. 48, Academic Press, Inc., pp. 384–409.

[12] Bern, M., and et. al, D. E. Emerging challenges in computational topology. In *Report from the NSF-funded Workshop on Computational Topology* (1999).

[13] Bern, M., and Plassmann, P. Mesh generaiton. In *Handbook of computational geometry* (1999), pp. 291–332.

[14] Bern, M. W., Eppstein, D., and Gilbert, J. R. Provably good mesh generation. In *IEEE Symposium on Foundations of Computer Science* (1990), pp. 231–241.

[15] Bloomenthal, J. *An Implicit Surface Polygonizer.* Academic Press, Boston, 1994.

[16] Bloomenthal, J. *Introduction to Implicit Surface: Surface Tiling.* Morgan Kaufmann, 1997.

[17] Board, O. A. R., Board, O. A. R., and Shreiner, D. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.4*, 4th ed. Addison-Wesley Professional;, November 2003.

[18] Boothby, W. M. *An Introduction to Differential Manifolds and Riemannian Geometry.* Academic Press, 1975.

[19] Borouchaki, Laug, P., and George, P.-L. Parametric surface meshing using a combined advancing-front generalized delaunay approach. In *International Journal for Numerical Methods in Engineering* (2000), vol. 49, pp. 233–259.

[20] Canann, S. A., Liu, Y.-C., and Mobley, A. V. Automatic 3d surface meshing to address today's industrial needs. In *Finite Elements in Analysis and Design* (1997), vol. 25, pp. 185–198.

[21] Cazals, F., Chazal, F., and Lewiner, T. Molecular shape analysis based upon the morse-smale complex and the connolly function. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry* (New York, NY, USA, 2003), ACM Press, pp. 351–360.

[22] Cheng, H. *Algorithm for Smooth and Deformable Surfaces in 3D.* PhD thesis, UIUC, 2002.

[23] Cheng, H., Dey, T. K., Edelsbrunner, H., and J.Sullivan. Dynamic skin triangulation. In *Discrete Comput. Geom.* (2001), vol. 25, pp. 525–568.

[24] Cheng, H., Edelsbrunner, H., and Fu, P. Shape space from deformation. In *Computational Geometry: Theory and Applications* (1998), vol. 19, pp. 191–204.

[25] Cheng, H., and Shi, X. Guaranteed quality triangulation of molecular skin surfaces. In *Proceedings of IEEE Visualization* (2004), pp. 481–488.

[26] Cheng, H., and Tan, T. Approximating polygonal objects by deformable smooth surfaces. In *Proceedings of Mathematical Foundations of Computer Science 2005* (2005), pp. 248–259.

[27] Cheng, S.-W., and Dey, T. K. Quality meshing with weighted delaunay refinement. In *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms* (2002), Society for Industrial and Applied Mathematics, pp. 137–146.

[28] Cheng, S.-W., Dey, T. K., Edelsbrunner, H., Facello, M. A., and Teng, S.-H. Sliver exudation. In *Proceedings of the fifteenth annual symposium on Computational geometry* (1999), ACM Press, pp. 1–13.

[29] Cheng, S.-W., Dey, T. K., Ramos, E. A., and Ray, T. Quality meshing for polyhedra with small angles. In *Proceedings of the twentieth annual symposium on Computational geometry* (2004), ACM Press, pp. 290–299.

[30] Chew, L. P. Constrained delaunay triangulations. In *SCG '87: Proceedings of the third annual symposium on Computational geometry* (New York, NY, USA, 1987), ACM Press, pp. 215–222.

[31] Connolly, M. L. Molecular surfaces: A review. In *Network Science* (1996).

[32] de Berg, M., Schwarzkopf, O., van Kreveld, M., and Overmars, M. *Computational Geometry: Algorithms and Applications*, 2nd ed. Springer-Verlag, 2000.

[33] de Cougny H.L., and M.S., S. Surface meshing using vertex insetion. In *Proceedings of 5th International Meshing Roundtable* (Oct. 1996), pp. 243–256.

[34] Delfinado, C. J. A., and Edelsbrunner, H. An incremental algorithm for betti numbers of simplicial complexes on the 3-sphere. In *Computer Aided Geometric Design* (November 1995), vol. 12, pp. 771–784.

[35] Dey, T. K., Giesen, J., Ramos, E. A., and Sadri, B. Critical points of the distance to an epsilon-sampling of a surface and flow-complex-based surface reconstruction. In *SCG '05: Proceedings of the twenty-first annual symposium on Computational geometry* (New York, NY, USA, 2005), ACM Press, pp. 218–227.

[36] Edelsbrunner, H. The union of balls and its dual shape. In *Proceedings of the ninth annual symposium on Computational geometry* (1993), ACM Press, pp. 218–231.

[37] Edelsbrunner, H. Smooth surfaces for multiscale shape representation. In *Proc. Sympos. Found. Software Techn. Theoret. Comput. Sci.* (1995), pp. 391–412.

[38] Edelsbrunner, H. Deformable smooth surface design. In *Discrete Computational Geometry* (1999), vol. 21, pp. 87–115.

[39] Edelsbrunner, H. Triangulations and meshes in computational geometry. In *Acta Numerica* (2000), pp. 133–213.

[40] Edelsbrunner, H. *Geometry and topology for mesh generation.* Cambridge University Press, New York, NY, USA, 2001.

[41] Edelsbrunner, H., Facello, M., and Liang, J. On the definition and the construction of pockets in macromolecules. Tech. rep., University of Illinois at Urbana-Champaign, 1995.

[42] Edelsbrunner, H., and Guoy, D. Sink-insertion for mesh improvement. In *SCG '01: Proceedings of the seventeenth annual symposium on Computational geometry* (New York, NY, USA, 2001), ACM Press, pp. 115–123.

[43] Edelsbrunner, H., Harer, J., Natarajan, V., and Pascucci, V. Morse complexes for piecewise linear 3-manifolds. In *Proceedings 19th ACM Symposium on Computational Geometry.* (2003), pp. 361–370.

[44] Edelsbrunner, H., Harer, J., and Zomorodian, A. Hierachial morse complexes for piecewise linear 2-manifolds. In *Proceedings 17th ACM Symposium on Computational Geometry.* (2001), pp. 70–79.

[45] Edelsbrunner, H., and Koehl, P. The geometry of biomolecular solvation. In *Discrete and Computational Geometry (MSRI Publications)* (2005), vol. 52, pp. 241–275.

[46] Edelsbrunner, H., Letscher, D., and Zomorodian, A. Topological persistence and simplification. In *Symposium on Foundations of Computer Science.* (2000), pp. 454–463.

[47] Edelsbrunner, H., Li, X.-Y., Miller, G., Stathopoulos, A., Talmor, D., Teng, S.-H., Ungor, A., and Walkington, N. Smoothing and cleaning up slivers. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing* (2000), ACM Press, pp. 273–277.

[48] Edelsbrunner, H., and Mucke, E. P. Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. In *ACM Trans. Graph.* (1990), vol. 9, ACM Press, pp. 66–104.

[49] Edelsbrunner, H., and Mucke, E. P. Three-dimensional alpha shape. In *ACM Transaction on Graphics* (Jan 1994), vol. 13, pp. 43–72.

[50] Edelsbrunner, H., and Shah., N. Triangulating topological spaces. In *Proceedings 10th ACM Symposium on Computational Geometry.* (1994), pp. 285–292.

[51] Edelsbrunner, H., and Shah, N. R. Incremental topological flipping works for regular triangulations. In *Symposium on Computational Geometry* (1992), pp. 43–52.

[52] Eisenberg, D., and McLachlan, A. Solvation energy in protein folding and binding. In *Science* (1986), vol. 319, pp. 199–203.

[53] Elcock, A. H., Sept, D., , and McCammon, J. Computer simulation of protein-protein interactions. In *Journal of Physical Chemistry* (2001), vol. 105, pp. 1504–1518.

[54] Erickson, J. Nice point sets can have nasty delaunay triangulations. In *Proceedings of the seventeenth annual symposium on Computational geometry* (2001), ACM Press, pp. 96–105.

[55] Farin, G. *Curves and surfaces for computer aided geometric design (3rd ed.): a practical guide.* Academic Press Professional, Inc., San Diego, CA, USA, 1993.

[56] Fomenko, A. T., and Kunii., T. L. *Topological Modeling for Visualization.* Springer- Verlag Tokyo, 1997.

[57] Giesen, J., and John, M. The flow complex: a data structure for geometric modeling. In *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms* (Philadelphia, PA, USA, 2003), Society for Industrial and Applied Mathematics, pp. 285–294.

[58] Hao, C., and Bishop, J. Delaunay triangulation for curved surfaces. In *Proceedings 6th International Meshing Roundtable* (Oct 1997), pp. 115–127.

[59] Hao, X., and Varshney, A. Efficient solution of poisson-boltzmann equation for electrostatics of large molecules. In *High-Performance Computing Symposium* (April 2004), pp. 71 – 76.

[60] Hartmann, E. A marching method for the triangulation of surfaces. In *The Visual Computer* (1998), vol. 14, pp. 95–108.

[61] Hearn, D., and Baker, M. P. *Computer graphics*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1986.

[62] Hilton, A., Stoddart, A., Illingworth, J., and Windeatt, T. Reliable surface reconstruction from multiple range images. In *Lecture Notes in Computer Science* (1996), vol. 1064, pp. 117–127.

[63] Holst, M., Baker, N. A., and Wang, F. Adaptive multilevel finite element solution of the poisson-boltzmann equation i. algorithms and examples. In *Journal of Computational Chemistry* (2000), vol. 21, pp. 1319–1342.

[64] ITO, Y., and NAKAHASHI, K. Advancing front surface triangulation based on cad data. In *Proceedings of the Japan Society for Aeronautical and Space Sciences* (Oct. 2000), pp. 755–758.

[65] Kaitin, K. Biotech products proliferate, but total development times lengthen. Impact Report 6, Tufts Center for the Study of Drug Development, 2001.

[66] Karkanis, T., and Stewart, A. High quality, curvature dependent triangulation of implicit surfaces. In *IEEE Computer Graphics and Application* (2001), vol. 2, pp. 60–69.

[67] Kavraki, L. E. Geometry and the discovery of new ligands. In *Algorithms for Robotic Motion and Manipulation (WAFR1996)* (1997), J.-P. Laumond and M. Overmars, Eds., A.K. Peters, pp. 435–448.

[68] Kleywegt, G., and Jones, T. Detection, delineation, measurement and display of cavities in macromolecular structures. In *Acta Crystallogr D Biol Crystallogr* (Mar 1994), vol. 50, pp. 178–85.

[69] Kruithof, N., and Vegter, G. Approximation by skin surfaces. In *Symposium on Solid Modeling and Applications* (2003), pp. 86–95.

[70] Kruithof, N., and Vegter, G. Triangulating skin surfaces. Tech. rep., Technical Report ECG-TR-244303-01, Rijksuniversiteit Groningen, 2003.

[71] Kruithof, N., and Vegter, G. Meshing skin surfaces with certified topology. Manuscript, 2004.

[72] Lawson, C. L. Software for c^1 surface interpolation. In *Mathematical Software III* (1977), pp. 161–194.

[73] Leach., A. R. *Molecular Modelling.* Longman, Harlow, England, 1996.

[74] Liang, J., Edelsbrunner, H., Fu, P., Sudhakar, P., and Subramaniam, S. Analytical shape computation of macromolecules: I. molecular area and volume through alpha shape. In *Proteins* (Oct 1998), vol. 33, pp. 1–17.

[75] Liang, J., and Subramaniam, S. Computation of molecular electrostatics with boundary element methods. In *Biophysical Journal* (1997), vol. 73, pp. 1830–1841.

[76] Little, J. J. Extended gaussian images, mixed volumes, shape reconstruction. In *Proceedings of the first annual symposium on Computational geometry* (1985), ACM Press, pp. 15–23.

[77] Matsumoto, Y. *An Introduction to Morse Theory.* American Mathematical Society, 2002.

[78] Miller, G. L., Talmor, D., Teng, S.-H., and Walkington, N. A delaunay based numerical method for three dimensions: generation, formulation, and partition. In *STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing* (New York, NY, USA, 1995), ACM Press, pp. 683–692.

[79] Morse, M. *The calculus of Variations in the Large.* The American mathematical society , New York, 1934.

[80] Nicholls, A., and Honig, B. A rapid finite difference algorithm, utilizing successive over-relaxation to solve the poisson-boltzmann equation. In *J. Comput. Chem.* (New York, NY, USA, 1991), vol. 12, John Wiley & Sons, Inc., pp. 435–445.

[81] Owen, S. A survey of unstructured mesh generation technology. In *Proceedings 7th International Meshing Roundtable* (Oct 1998), pp. 239–267.

[82] Paule, M., and Gascuel, C. Layered deformable models with implicit surfaces. In *Graphics Interface* (1998 June), pp. 201–208.

[83] Petrie, G., and Kennie, T. J. M. Terrain modelling in surveying and civil engineering. In *Comput. Aided Des.* (Newton, MA, USA, 1987), vol. 19, Butterworth-Heinemann, pp. 171–187.

[84] Rocchia, W., Sridharan, S., Nicholls, A., Alexov, E., Chiabrera, A., and Honig, B. Rapid grid-based construction of the molecular surface and the use of induced surface charge to calculate reaction field energies: Applications to the molecular systems and geometric objects. In *Journal of Computational Chemistry* (June 2001), vol. 23, pp. 128 – 137.

[85] Rogers, D. F. *An Introduction to NURBS: With Historical Perspective.* Mogan Kaufmann, 2000.

[86] Rosen, M., Lin, S. L., Wolfson, H., and Nussino, R. Molecular shape comparisons in searches for active sites and functional similarity. In *Protein Engineering* (1998), vol. 11, pp. 263–277.

[87] Ruppert, J. A delaunay refinement algorithm for quality 2-dimensional mesh generation. In *SODA '93: Selected papers from the fourth annual ACM SIAM symposium on Discrete algorithms* (1995), Academic Press, Inc., pp. 548–585.

[88] Saupe, D., and Vranic, D. V. 3d model retrieval with spherical harmonics and moments. In *Proceedings of the 23rd DAGM-Symposium on Pattern Recognition* (2001), Springer-Verlag, pp. 392–397.

[89] Seveno, E. Towards and adaptive advancing front method. In *Proceedings 6th International Meshing Roundtable* (Oct 1997), pp. 349–360.

[90] Shewchuk, J. R. *Delaunay refinement mesh generation.* PhD thesis, Carnegie-Mellon University, 1997.

[91] Shewchuk, J. R. What is a good linear element? interpolation, conditioning, and quality measures. In *11th International Meshing Roundtable, Sandia National Laboratories* (September 2002), pp. 115–126.

[92] Stander, B. T., and Hart, J. C. Guaranteeing the topology of an implicit surface polygonization for interactive modelling. In *Proceeding SIGGRAPH 97* (August 1997), pp. 279–286.

[93] Tate, J. *Structural Bioinformatics.* Wiley-Liss, Inc., 2003, ch. Molecular Visualization, pp. 135–158.

[94] Taylor, R., Jewsbury, P., and Essex, J. A review of protein-small molecule docking methods. In *Jouranl of Computer-Aided Molecular Design* (2002), vol. 16, pp. 151–166.

[95] Teng, S.-H., and Wong, C. W. Unstructured mesh generation: Theory, practice, and perspectives. In *Int. J. Computational Geometry and Applications* (2000), vol. 10, pp. 227–266.

[96] Varshney, A., Brooks, F. P., William, J., and Wright, V. Linearly scalable computation of smooth molecular surfaces. In *IEEE Computer Graphics and Applications* (1994), vol. 14, pp. 19–25.

[97] Vigo, M., Pla, N., and Brunet, P. Curvature adaptive triangulations of surfaces. In *European Congress on Computational Methods in Applied Sciences and Engineering* (2000).

[98] Welch, W., and Witkin, A. Free-form shape design using triangulated surfaces. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1994), ACM Press, pp. 247–256.

[99] Zhang, Y., Xu, G., and Bajaj, C. Quality meshing of implicit solvation models of biomolecular structures. Ices technical report, University of Texas at Austin, 2004.

[100] Zhao, Y., Ong, H.-Y., Tan, T.-S., and Xiao, Y. Interactive control of component-based morphing. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 339–348.