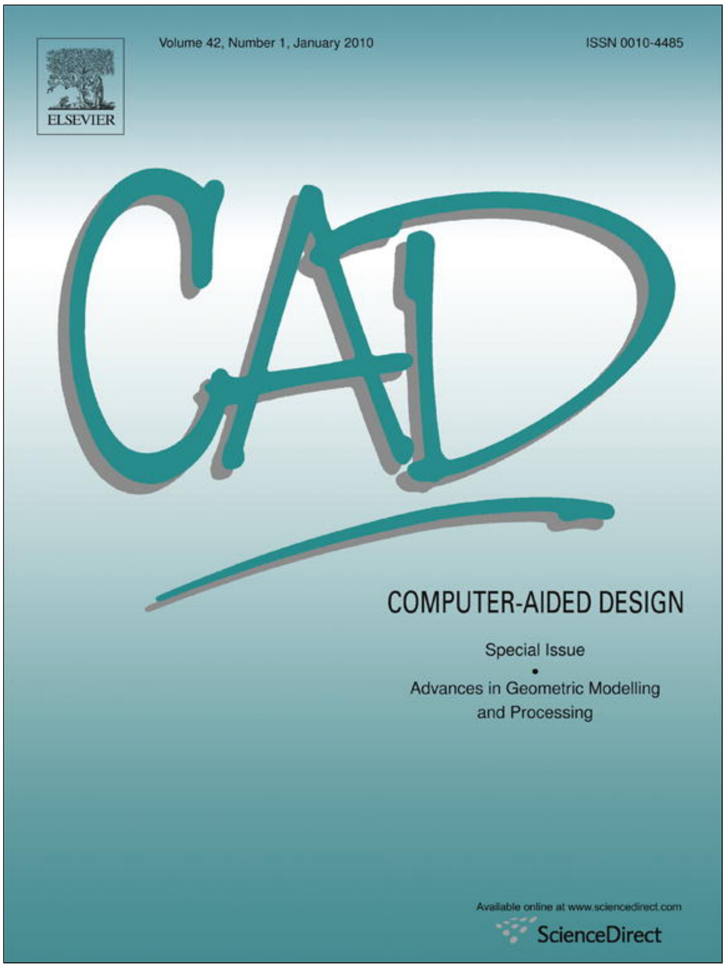
Provided for non-commercial research and education use.
Not for reproduction, distribution or commercial use.



This article appeared in a journal published by Elsevier. The attached copy is furnished to the author for internal non-commercial research and education use, including for instruction at the authors institution and sharing with colleagues.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

http://www.elsevier.com/copyright

Author's personal copy

Computer-Aided Design 42 (2010) 67–75

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Carving for topology simplification of polygonal meshes

Nate Hagbi, Jihad El-Sana *

Computer Science Department, Ben-Gurion University, Israel

ARTICLE INFO

Article history:
Received 15 July 2008
Accepted 2 April 2009

Keywords:
Model simplification
Topology simplification
Level-of-detail generation
Shape approximation
Geometric modeling

ABSTRACT

The topological complexity of polygonal meshes has a large impact on the performance of various geometric processing algorithms, such as rendering and collision detection algorithms. Several approaches for simplifying topology have been discussed in the literature. These methods operate locally on models, which makes their effect on the topology hard to predict and analyze. Most existing methods tend to exhibit several disturbing artifacts, such as shrinking of the input and splitting of its components. We propose a novel top-down approach for topology simplification that avoids most problems that are common in existing methods. We start with a simple, genus-zero mesh that bounds the input and gradually introduce topologic features by a series of carving operations. This process yields a multiresolution stream of meshes with increasing topologic level of detail. We further present a carving algorithm that is based on constrained Delaunay tetrahedralization. The algorithm first constructs the tetrahedral mesh of the complement of the input with respect to its convex hull. It then proceeds to eliminate tetrahedra in a prioritized manner. We present quality results for two families of meshes that are difficult to simplify by all existing methods known to us: topologically complex meshes and highly clustered meshes.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction

Topology simplification has been identified in the graphics literature as a critical task for various general and domain-specific applications. Simplifying the topology of complex models is often necessary for geometric simplification to achieve quality results. Topology simplification might also be necessary for generating level-of-detail representations. For this purpose, aggregating a large number of objects or components might be required before simplifying each of the objects thoroughly. Topology simplification is important for efficiency reasons as well. The performance of collision-detection algorithms is known to depend heavily on the genus of objects [1]. Finally, topologic features can cause several undesirable artifacts, such as image-space aliasing. It is thus desirable to simplify the topology of polygonal models before or through geometric simplification.

Given a mesh, we are interested in generating a topological multiresolution stream of approximations. The component count and genus of the approximations will be represented with increasing detail through the stream. We would also like the geometric complexity of the mesh to be monotonic in the number of vertices in order to provide progressively refined geometric approximations. Another important property we require from the algorithm is to produce approximations that are guaranteed to be free of self-intersection.

Several approaches have been proposed in the literature to reduce the topologic complexity of polygonal meshes. In most of these approaches, the simplification process affects the topology in a way that is hard to predict and analyze. Proximity-based methods, such as pair-collapse [2,3] and clustering [4,5], are based on the proximity between vertices. They allow topological changes and aggregation but do not identify and remove parts of the surface that become internal. Thus, they may end up complicating the topology rather than simplifying it. Such methods can also join components or do the opposite (see Fig. 1). The number of components in the model is then not monotonic, as it can increase and decrease unexpectedly. In contrast, one would like each component of the original surface to be represented by at most one component in each generated approximation. In general, the number of components is preferred to be monotonically decreasing or increasing through the decimation or refinement, respectively.

Shrinking of the input is another problem that is common in most existing simplification methods. In addition to a reduction in the overall size of the model, shrinking causes small components to reach sub-pixel dimensions and finally disappear. A class of meshes particularly vulnerable to shrinking consists of meshes whose center is highly clustered, i.e., consists of a large number of small components. In such models the vertices in the center pull

* Corresponding address: Computer Science Department, Ben-Gurion University of the Negev, P.O.B 653, 84105 Be'er Sheva, Israel. Tel.: +972 8647 7871; fax: +972 8 6477650.

E-mail addresses: natios@cs.bgu.ac.il (N. Hagbi), el-sana@cs.bgu.ac.il (J. El-Sana).

0010-4485/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cad.2009.04.001

Author's personal copy

Fig. 1. The effect of the pair-collapse operator on a topologically complex model and a highly clustered non-manifold model: (a) Original models, (b) pair-collapse simplified models. Note the increased number of disconnected components and shrunk appearance.

the approximation towards the inside. In contrast, when viewing from the outside, one would like a highly clustered model to be coarsened from the inside towards the outside. This can maintain the general shape and dimensions of the model. It can also be motivated by a basic assumption that usually the central part of a complex model is less visible to an outside viewer. For instance, to simplify a model of a detailed motor that includes all the internal mechanical parts, it is reasonable to begin with a more aggressive simplification on the inside than on the outside.

This work presents a novel approach for topology simplification of unstructured polygonal surfaces. Our method addresses all the above issues and avoid problems that are common in most topology simplification schemes. We simulate the action of a carver and draw from the α-shape [6] and shrink-wrapping [7] concepts. Using a top-down refinement scheme, we construct a multiresolution representation that coarsens topologically. We start with a genus-zero bounding object, such as the convex hull, as an initial approximation and gradually introduce topologic features by splitting to components and creating tunnels. We achieve this by tetrahedralizing the complement of the input with respect to the initial approximation, and carving the resulting tetrahedra from the initial approximation until the input is reached. To control the order, we prioritize the tetrahedra according to the error they introduce into the model. We further propose two parameterizations to the algorithm for controlling its effect and avoiding two problems that can arise, referred to as snaking and spiking. We demonstrate that the proposed method allows both aggregation of multiple components and genus reduction in a natural way, while maintaining the visual properties of meshes.

The remainder of this paper is organized as follows. Section 2 reviews related topology simplification approaches, as well as surface reconstruction and remeshing concepts. Section 3 outlines our approach and discusses design decisions regarding the implementation of a carving algorithm. Section 4 describes our carving algorithm, which is based on constrained Delaunay tetrahedralization (CDT) [8,9]. Section 5 describes two useful parameterizations of the algorithm that provide control over the process and avoid problems in produced approximations. Section 6 presents some results and discussion. Section 7 summarizes our contribution, and describes limitations and future work.

2. Related work

In this section we briefly review related work from the field of topology simplification, as well as concepts from surface reconstruction and remeshing.

2.1. Topology simplification

In various cases, topology simplifying steps have to be taken for performing aggressive geometric simplification. This can be necessary in order for the input not to degenerate or lose its basic characteristics. Meshes consisting of multiple geometrically disjoint components or scenes consisting of a large number of objects demand a different approach than simplifying each component separately. In such cases, even after each component has reached minimum resolution, the number of components can still imply a massive model size. The problem we address is thus the simplification of a topologically complex model, which consists of multiple disconnected components and high genus.

Rossignac and Borrel [5] propose imposing a global grid over the input dataset and cluster vertices into cells, which are then replaced by unified vertices. Later, Luebke and Erikson [4] extended this method and defined a tight octree over an input model to generate a view-dependent hierarchy. These approaches can reduce the genus and number of components if the desired simplification regions fall within a grid cell. However, it can also widen holes and tunnels that fall across grid cells.

He et al. [10] proposed using a low-pass filter in the volumetric domain to simplify the genus of the input in a controlled fashion. This method is capable of aggregating disjoint components and reducing the genus of objects. However, there is no correlation between vertices of the input and vertices of the simplified mesh. In addition, meshes simplified by this method are subject to shrinking. Wood et al. [11] developed an algorithm for removing topological errors in isosurfaces in the form of tiny handles by sweeping the models and constructing a Reeb graph. Bischoff et al. [12] extract isosurfaces with controlled topology. They start with an initial estimate of the final surface and then morph the model into the final result using a topology-preserving growing scheme. Zhou et al. [13] recently proposed a solid modeling approach that repairs topological errors in the form of small surface handles. Gerstner and Pajarola [14] proposed a technique for controlled topology simplification by multiresolution isosurface extraction. Andujar et al. [15] have developed an automatic simplification algorithm to generate a multiresolution representation from an initial boundary representation of a polyhedral solid. However, this method is not straightforward to implement and produces non-regular regions. These methods work in the volumetric domain and perform topology simplification by converting a three-dimensional (3D) polygonal model to and from its volumetric representation. This conversion usually does not preserve the correspondence between the vertices of the input model and the vertices of the simplified one. It also often results in an increased

Author's personal copy

complexity related to the volume of the object, rather than to its original representation.

Popović and Hoppe [3] and Garland and Heckbert [2] proposed the vertex pair-collapse operator, which can be considered the topology modifying variant of the edge-collapse operator. Pair-collapse based algorithms can join components of a model together and therefore operate much better than edge-collapse on complex inputs consisting of numerous components [2]. However, the pair-collapse operator performs topology altering steps without analyzing the resulting topology and can demonstrate several undesirable artifacts. For instance, pair-collapse does not keep track of parts of the surface that become internal as a result of topology altering steps. Moreover, though capable of aggregating disjoint components, the pair-collapse operator can also do the opposite — divide a single component (see Fig. 1). Schroeder [16] introduced the vertex-split operator, which is designed to keep the mesh consistent when a topology simplifying vertex-decimation step takes place and a non-manifold neighborhood of a vertex is formed. However, vertex-decimation is still unable to aggregate disjoint components and does not identify and remove parts of the surface that become internal.

El-Sana and Varshney [17] propose rolling a sphere of radius α over the surface of the input mesh. Areas not accessible to the sphere are identified and removed and the resulting holes are resurfaced. This work does not suggest any way of aggregating disjoint components. In addition, smoothly resurfacing the removed parts of the surface is an intricate task. This makes the method suitable for objects with tunnels, holes and cavities, which are otherwise flat, such as mechanical parts, rather than smoothly curved, such as animal models. This operation also produces a single approximation per execution, and suggests no intuitive way for automatically generating a multiresolution stream. Ju et al. [18] presented a guided topology editing algorithm for geometric models. The resulting model is topologically consistent with a defined target shape. The algorithm allows removing and adding topologic features, while ensuring that each change is performed by minimal modification to the source model.

2.2. Surface reconstruction and remeshing

Two fields that are closely related to our work are surface reconstruction and remeshing. Surface reconstruction methods are commonly used to reconstruct a scanned physical object from range scanned data points. Remeshing algorithms convert general meshes into meshes with favorable properties, such as subdivision connectivity.

Edelsbrunner and Mücke [19] defined the 3D α-shape over the Delaunay tetrahedralization of a given set of points. Amenta and Bern [20] proposed the first surface reconstruction algorithm with correctness guarantee under a given sampling condition. Bernardini and Bajaj [21] proposed using α-hulls to elegantly reconstruct a surface from an unorganized set of points that describes a mechanical model. Attene and Spagnuolo [22] proposed constructing the Delaunay tetrahedralization of the point set and then performing a sculpturing process. Gopi et al. [23] developed a projection-based approach to determine the faces incident to a point and used sampling criteria to guarantee a topologically correct mesh after surface reconstruction. Recently, Giesen and John [24] extended the Gabriel graph by associating its edges with the critical points of a dynamic system induced by the sample points. The above approaches start with a set of points and aim at reconstructing the appropriate surface, while our approach starts with a surface and aims to generate topology-simplified representations for it. The quality of the result of the above reconstruction methods is measured with respect to the distance of the reconstructed surface from the input point set, while the quality of the result of topology simplification algorithms is measured with respect to the input surface. Kobbelt et al. [7] simulate a shrink-wrap process to convert a given unstructured triangle mesh into a mesh with subdivision connectivity. The complexity of the base meshes generated by this method is relatively small. However, this approach only handles genus-zero objects and does not perform topology-altering operations.

3. Geometric carving

We propose to perform topological refinement by a top-down scheme. We consider the input model to be the objective of the process and start with a coarse initial approximation that bounds it. By a series of carving operations we gradually introduce details into the initial approximation. Through each such carving operation we eliminate a part of the complement of the input with respect to the initial approximation. Fig. 2 illustrates an objective (f) gradually carved from its convex hull (a). In each iteration, a piece of the representation is removed to expose model details.

Carving can be applied to meshes in two or three dimensions. We describe the method for 3D meshes and assume that the application for 2D meshes is straightforward. Given a mesh M, we first construct a coarse genus-zero mesh C_0 bounding M (for example, the convex hull of M). We then construct a tetrahedralization T of the complement of M with respect to C_0, C_0/M, in a fashion constrained by the faces of C_0/M.[1] Next, we iteratively remove the tetrahedra of T from the current approximation until the objective M is reached. The tetrahedron elimination step is carried out by a carving operator, which takes the mesh C_i and a tetrahedron $t \in T$ as input and performs the required update operations to remove the tetrahedron t from C_i. This effectively turns the volume of t into a volume external to the approximation C_i. Through this process, we produce a chain of approximations, in which C_0 is the first and coarsest. Each application of the carving operator leads to a finer approximation of M. Let $C_i \xrightarrow{t} C_{i+1}$ denote that C_{i+1} is obtained by carving the tetrahedron t from C_i. A given order for the tetrahedra of T defines a chain of approximations for M,

$$C_0 \xrightarrow{t_0} C_1 \xrightarrow{t_1} \cdots \xrightarrow{t_{n-2}} C_{n-1} \xrightarrow{t_{n-1}} C_n = M, \quad (1)$$

where t_i is the ith tetrahedron carved and $n - 1$ is the number of tetrahedra in T. The key design decisions to be made are therefore about the way of constructing the initial approximation, the manner of tetrahedralizing the complement and the order of removing the tetrahedra.

We would like to carve tetrahedra that share a face with the surface of an approximation before internal tetrahedra. Otherwise, internal voids will be formed, which turn a mesh from a manifold one to a non-manifold one. For this purpose, let us first consider several definitions. A face of an approximation is said to be a constraining face if it belongs to M. The degree of a tetrahedron t with respect to an approximation C and an input mesh M, $d_{C,M}(t)$, is defined as the number of faces of t that are on the boundary of C and do not belong to M. The possible degrees of a triangle in the 2D case are depicted in Fig. 3.

We define a tetrahedron $t \in T$ to be *carvable* from a mesh C_i if $d_{C_i,M}(t) > 0$, i.e., if at least one of its non-constraining faces adjoins the exterior domain of C_i, i.e., the tetrahedron is on the boundary between the inside and outside of C_i. Equivalently, t is carvable from C_i if either t has a non-constraining face that

[1] We assume that the complement is tetrahedralizable. Otherwise, we can introduce Steiner vertices to tetrahedralize it [25].

Author's personal copy

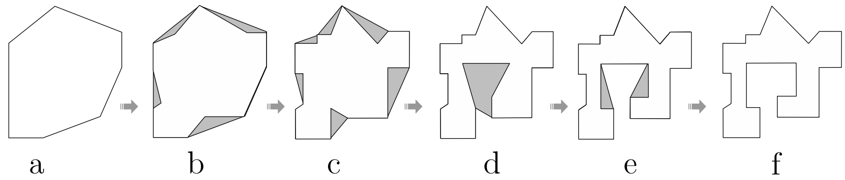
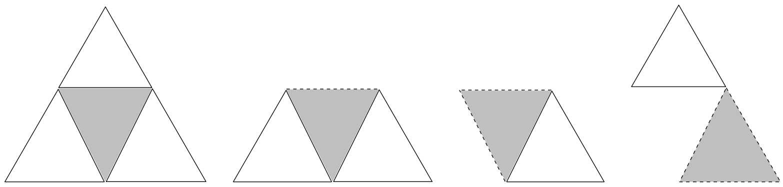


Fig. 2. The carving process.



(a) $d_{C,M}(t) = 0$. (b) $d_{C,M}(t) = 1$. (c) $d_{C,M}(t) = 2$. (d) $d_{C,M}(t) = 3$.

Fig. 3. The possible degrees of a shaded triangle in two dimensions.

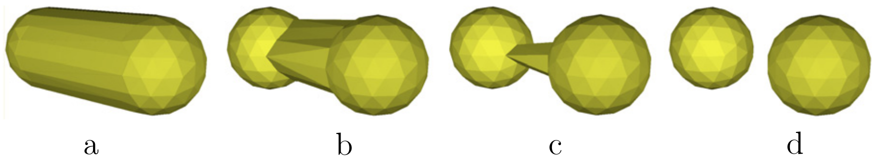


Fig. 4. Two spheres splitting through the carving process.

adjoins the exterior domain of C_0 or one of the tetrahedra sharing a non-constraining face with t has been carved through the $i - 1$ previous steps. We shall consider in each iteration only the carvable tetrahedra in T.

Notice that by the above definition of a carvable tetrahedron, meshes that contain internal empty volumes (internal regions that do not belong to the model) are not refined by the algorithm to include these volumes, since the tetrahedra that fill them are never marked as carvable. To carve these volumes, we mark one face in each such isolated void as a non-constraining face that adjoins the exterior of the approximation, effectively increasing its tetrahedron's degree. This causes a single tetrahedron in each empty volume to be carved.

Using the above refinement scheme, topological features like tunnels are naturally formed, rather than eliminated. This process also naturally fuses multiple disconnected components together when forming the genus-zero initial approximation. The initial approximation is then a single object that splits throughout the process into separate components, as the parts of the complement that connect them are eliminated. Fig. 4 depicts the effect of carving on a model of two spheres. As tetrahedra in the area between the spheres are gradually carved, the spheres naturally split. In addition, the number of vertices and faces required to represent the input gradually increases as detail is added, since the vertices and faces of the input mesh that are internal to the approximation and have not been introduced yet are not included.

If we consider the resulting multiresolution chain reversely, we begin with M and converge to the initial approximation C_0. The image-space area that is bounded by the silhouette of any approximation in the chain, from any viewpoint, contains the area bounded by the silhouette of the input. The approximation monotonically expands within the boundaries of the initial approximation. This property prevents model parts from disappearing by reaching sub-pixel dimensions.

3.1. Initial approximation

The initial mesh C_0 and the objective M form two opposite extremes for a range of approximations constructed through the process, since any approximation generated is contained by C_0 and contains M. C_0 is thus the coarsest approximation produced by the algorithm, which implies that, in order to achieve maximal simplification of topology, C_0 should be a single component, genus-zero object.

The bounding-box of M seems to be, at first glance, a good initial approximation. The bounding-box of M is genus-zero, of constant complexity, and can be calculated easily in linear time with respect the number of vertices in the model. Unfortunately, the vertices of the bounding-box can lie far from any vertex of M, and can thus form features that hardly depend on M. Moreover, vertices of the bounding-box tend to receive a relatively high degree of tetrahedralizations of the complement that do not introduce additional Steiner vertices into the faces of the bounding-box (see Fig. 5(a)). This leads to unpleasant artifacts in the approximations, such as sharp features.

The fairly opposite extreme for the choice of initial approximation is the convex hull of M, which consists solely of vertices of M and contains it (see Fig. 5(b)). A problem with using the convex hull as an initial approximation is that parts of the input model M that reside on its convex hull start the process fully refined, which implies that they are not simplified in the multiresolution stream. This problem can be handled in several ways and will be addressed shortly. For the results presented in this paper, we used the convex hull of M as the initial approximation. Note that in the case of neighboring coplanar faces, the triangulation, and thus the structure of the convex hull, will depend on the convex hull algorithm implementation used.

3.2. Complement constrained tetrahedralization

The complement can be tetrahedralized in advance to determine the pieces that ought to be removed in order to extract the input from the initial approximation. Although the tetrahedralization can be performed implicitly by local carving operations, pretetrahedralization assists in clarifying the concept and still achieves quality results. The tetrahedralization method is of high importance to the results of a carving algorithm, since it determines the characteristics of the tetrahedra to be carved and

Author's personal copy

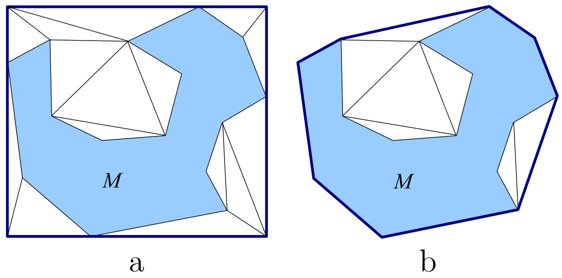


Fig. 5. Triangulation of the complements of M with respect to (a) the bounding-box and (b) the convex hull of M.

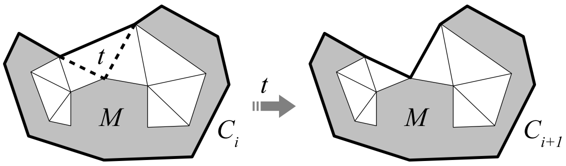


Fig. 6. The result of the carving operator applied to a triangle t and a mesh C_i, yielding C_{i+1}.

consequently the geometry and topology of the generated approximations.

An important feature of a tetrahedron that determines the visual change it introduces is its circumradius-to-shortest-edge ratio, which is the ratio between the radius of the surrounding sphere and the shortest edge of the tetrahedron. Shewchuk [26] defines a d-simplex as *skinny* if its circumradius-to-shortest-edge ratio is larger than some constant threshold to decide whether it should be eliminated from the triangulation. The effect of carving a skinny tetrahedron is expected to be smaller and less perceptible than the effect of carving a fat tetrahedron of roughly the same volume. Tetrahedralizations that consist of skinny tetrahedra result in slow development of the approximation throughout the process. Carving fat tetrahedra, on the other hand, yields rough and deep cavities, which can refine the approximation rapidly. Another feature that determines the effect of the carving operator on a mesh is the volume of the carved tetrahedron. Tetrahedra with larger volumes introduce larger effects into the approximation. Note that as our approach is based on a refinement scheme, we would like the process to converge rapidly towards the objective. These two features of tetrahedra motivate our choice of constrained Delaunay tetrahedralization (CDT) (see Section 4).

3.3. Carving operator

The role of the carving operator is to perform a set of updates to reflect a single carving operation to the surface. The effect of carving a triangle is illustrated in Fig. 6 on a manifold 2D mesh. Note that the surface of the approximation is denoted by bold lines before and after the operation. It is easy to see that the carving process is monotonic in the number of vertices, since vertices that are introduced into the approximation participate in all the subsequent approximations. Carving is also monotonic in the number of components, since the components are never joined, and can only be split. We take here a simple subset-placement strategy, which means that the vertices of any simplified mesh form a subset of the vertices of the original mesh. We remove a set of non-constraining faces from the mesh and introduce a set of new faces.

Let us describe more precisely the behavior of the operator for a mesh in R^d. Let t be a d-simplex, which is carvable from a mesh C, let F_{ext} be the set of hyperfaces of t that adjoin the exterior of C, and let F_{Mext} be the set of constraining hyperfaces in F_{ext}, i.e., $0 \leq |F_{Mext}| < |F_{ext}| \leq d+1$. The carving operator eliminates the non-constraining hyperfaces of t in F_{ext} from C, and introduces the $d+1-|F_{ext}|+|F_{Mext}|$ remaining hyperfaces of t into C, as well as possibly a single new vertex of t. While in a manifold mesh, and thus closed complement, constraining hyperfaces cannot simultaneously belong to a carvable d-simplex and adjoin the exterior domain, the situation is common in non-manifold meshes. In addition, for non-manifold meshes, constraining hyperfaces are introduced along with their opposite-facing hyperfaces—their folds. Fold hyperfaces can easily be tracked and eliminated.

Note that by itself a carving operation can turn a manifold mesh into a non-manifold one. A non-manifold neighborhood of a vertex can be formed, for instance, when a tunnel is opened by carving two tetrahedra from two opposite sides, or when two components split. This problem does not influence in any way the operation of the carving algorithm, which makes no assumption about connectivity and operates solely on tetrahedra. Therefore, we only need to solve the problem when outputing approximated surfaces. We adopt the method described in [27] to convert the approximated mesh to a manifold surface.

Let us consider the case in which $d_{C,M}(t) = 1$, i.e., a single non-constraining hyperface s of t adjoins the exterior of C, and assume carving t introduces a new vertex, $v \in t, v \notin s$. The carving operator removes the hyperface s from the mesh and introduces the d remaining hyperfaces, which are adjacent to the new vertex v in t. Carving t from C in this case is equivalent to inserting a Steiner point inside s and contracting it to v. Note that only d-simplices of degree 1 may introduce a new vertex into the approximation, since, in all other cases, the $d+1$ vertices of the d-simplex already exist in the approximation.

If we reverse the basic carving operator to add d-simplices of degree 1 to the approximation, instead of carving them, we get a vertex-decimation operator, in which a single vertex of degree d is removed along with the hyperfaces in its neighborhood, and the resulting hole is retriangulated. Since the hole consists of exactly d vertices, the triangulated surface is a hyperface, consisting of the d vertices of the resulting hole.

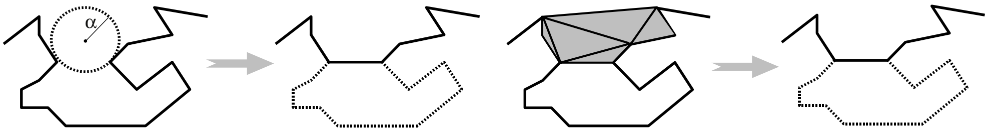
3.4. Exposed tetrahedra

We introduce a heuristic that reduces the number of faces required for representing approximations, without changing the number of vertices. We aim at taking advantage of each vertex added as much as possible for carving the complement. As mentioned above, only tetrahedra of degree 1 can introduce a new vertex into the approximation, while tetrahedra of higher degree consist of vertices that have already been introduced. Notice also that carving tetrahedra of degree higher than 1 does not increase the number of faces in the approximation. For instance, carving a tetrahedron of degree 2 is equivalent to flipping the exposed edge of the tetrahedron. We thus define a tetrahedron t to be an *exposed* tetrahedron of C with respect M, if $d_{C,M}(t) > 1$. Namely, a tetrahedron is exposed when more than one of its faces adjoins the exterior domain of the current approximation. Through the carving process when a tetrahedron is carved its neighbors are checked. If any of the neighbors is found to be exposed, it is carved as well, its neighbors are processed in the same manner, and so on.

4. A CDT carving algorithm

Our algorithm is based on constrained Delaunay tetrahedralization (CDT) [8,9] for dividing the complement. The CDT maintains several favorable properties, such as low circumradius-to-shortest-edge ratio of tetrahedra. This allows reducing the effect of skinny tetrahedra, which introduce additional vertices and

Author's personal copy



(a) The effect of a shrinking alpha-ball. (b) The effect of carving.

Fig. 7. The effects of rolling a shrinking alpha-ball and carving.

faces but contribute little to the refinement of the mesh in the multiresolution stream. Skinny tetrahedra that do exist in the CDT are usually flat with constraining faces that occlude most of the visibility within the circumsphere of the tetrahedron. Open spaces are regularly tetrahedralized by relatively round tetrahedra.

The CDT carving algorithm first constructs the CDT for the complement and then begins to carve its tetrahedra until the objective is reached. We use three main data structures. The first is a CDT data structure that maintains the connectivity between tetrahedra. We maintain the current approximation separately and update it in each step according to the operations performed on the CDT. In addition, we use a priority queue to manage the order of the tetrahedra, where at each stage tetrahedra that become carvable are placed in the queue.

Our algorithm has an effect similar to the effect of rolling a sphere with decreasing radius, as proposed in [17]. Fig. 7 demonstrates the effects of the genus reduction method in [17] and our method. Notice that tuning the longest-edge length of the tetrahedra carved in each step is somewhat similar to tuning the radius of a sphere rolled over the surface of the mesh. When one decreases the radius of the sphere, wider spaces of the complement of the input with respect to its convex hull are opened before narrower spaces. Similarly, we attempt to carve tetrahedra that fill the widest spaces at each stage of the refinement process. This way, wide tunnels will be introduced before narrow tunnels. For this purpose, we experimented with several measure functions for prioritizing tetrahedra based on two terms — the volume and the longest-edge length. The longest-edge length term helps avoiding artifacts caused by the volume measure, such as sharp features. These result from the volume measure being discontinuous over neighboring tetrahedra. For the results presented in this paper we used the longest-edge length measure.

The worst-case time complexity analysis of the algorithm for a 3D mesh is as follows. The extraction of the complement can be performed by checking for each face of C_0 whether it exists in M and vice versa. This can be implemented using a hash table in time $O(c\log(c))$, where c is the maximum between the number of faces in M and the number of faces in C_0. The time complexity of the convex hull construction step is $O(v\log(v))$, where v is the number of vertices in M. Using the sweep-plane algorithm by Shewchuk [9], one can construct the CDT of the complement in time $O(vs)$, where s is the number of tetrahedra in the CDT. Constructing the queue from the CDT is performed in $O(s\log(s))$ time. This is also the total time complexity of the carving steps through an algorithm execution. Since $v \leq s$ and $c \leq s$, the overall time complexity of the algorithm is $O(vs + s\log(s))$.

5. Carving style

In this section we describe two parameterizations for the proposed algorithm: carving depth and breadth. These provide control over the behavior of the algorithm, and allow avoiding problems that can arise. Such problems are also common in surface reconstruction algorithms based on shrink-wrapping. Although the quality measure used for prioritizing tetrahedra balances the carving effect, in some cases undesired behavior can occur. Two of these problems and our treatment in these cases are described next.

5.1. Snaking

The snaking problem occurs when the algorithm carves a narrow and deep set of tetrahedra that go below the surface of the approximation, as shown in Fig. 9. These tetrahedra are usually not visible from an outside viewpoint and contribute little to refining the mesh. This problem partially results from the localness of the metric used to prioritize tetrahedra. The longest-edge length measure, for example, does not take into account the depth of a tetrahedron from the initial surface.

5.2. Spiking

Poor tetrahedralization may also lead to spiking of the surface. This occurs when the algorithm carves a skinny tetrahedron with a relatively small portion visible on the surface. It results in a narrow and deep cavity, as depicted in Fig. 9. Like snaking tetrahedra, a spiking tetrahedron contributes little to the refinement and may result in artifacts.

5.3. Carving depth

We use depth parameterization to control the way the algorithm advances towards the interior of the approximation, by balancing between the original quality measure described above and the depth of a tetrahedron. We achieve this by modifying the quality measure to take into account the depth of a tetrahedron from the surface of the initial approximation. The depth can be defined in different ways. One measure we used is the distance of the centroid of the tetrahedron from the initial surface. We can also define the depth of a tetrahedron with respect to an execution of the algorithm in a hierarchical manner: the level of a tetrahedron with a face that adjoins the exterior domain of the initial approximation is defined to be 1. When carving a tetrahedron, neighboring tetrahedra that become carvable as a result of the operation receive the depth of the tetrahedron incremented by 1. This way, the depth is defined only for tetrahedra that become carvable and calculated when they become so. The priority of a tetrahedron is modified to be inversely proportional to its depth. We divide the priority function by a power of the depth of a tetrahedron, t as shown in Eq. (2), where $q(t)$, $d(t)$ and dp are the priority, depth and depth parameter.

$$q_d(t) = q(t) \cdot d(t)^{-dp}. \quad (2)$$

For simplified illustration, consider the model of eight geospheres depicted in Fig. 10(d). The convex hull of the model is the chamfered box in Fig. 10(a). Tetrahedra connecting pairs of geospheres that share an edge in the box are skinny tetrahedra with poor ratio, while tetrahedra away from the edges of the box are rounder and larger. As a result, carving the model with no depth parameterization would eliminate the tetrahedra towards the center of the box through the centers of its faces before affecting the tetrahedra between geospheres (see Fig. 10(b)). This results in a genus-six object, in which the original geospheres are connected by skinny tetrahedra. The use of the depth parameter value balances the operation by decreasing the priority of the tetrahedra in the center and achieving a more uniform effect (see Fig. 10(c)).

Author's personal copy

(a) 250 vertices. (b) 700 vertices. (c) 2154 vertices.

Fig. 8. Topology-simplified levels of detail for a foot bone model.

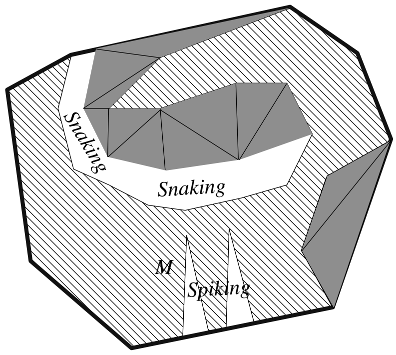


Fig. 9. Snaking and spiking due to poor tetrahedralization of the complement. The hatched and dark regions depict the original model and its complement, respectively. White regions illustrate carved parts, demonstrating the snaking and spiking artifacts.

5.4. *Carving breadth*

We can control the area on the surface of the approximation affected by a carving operation. Instead of carving a single tetrahedron in each step, we carve in one step a set of neighboring tetrahedra. We impose a sphere with radius *bp* on the centroid of a tetrahedron in each step, and carve all the tetrahedra whose center lies within the sphere and are reachable via adjacency traversal. By tuning the radius of the sphere we can control the breadth of each carving step. Consider, for example, the Yucca group of leaves model in Fig. 11(d), whose convex hull appears in Fig. 11(a). The elongated structure of the leaves causes skinny tetrahedra to be formed. Fig. 11(b) shows the artifacts that result from carving these tetrahedra one by one. Fig. 11(c) shows the effect of carving sets of tetrahedra. By maintaining the adjacency list for each tetrahedron, determining the neighboring tetrahedra within an imposed sphere can be performed in $O(s)$, adding a total time complexity of $O(s^2)$ to the algorithm in the worst case, where s is the total number of tetrahedra in the CDT.

6. Results

In this section we describe our results and running times. Fig. 13 demonstrates the effect of the CDT carving algorithm on a detailed torso model. The ribs of the torso form complex tunnels, which cannot be simplified using topology-preserving methods and are difficult to simplify using existing topology simplifying operators, such as pair-collapse (as demonstrated in Fig. 1). The same difficulty holds for the sconce model in Fig. 12. The tunnels of different sizes and shapes are sealed by the CDT carving algorithm, due to the properties of tetrahedra in the area. Fig. 13(d) presents a simplified version of the torso, which is visually almost identical to the original due to the excessive detail of the spine in the original model. The genus of the torso remains low until the approximation acquires the shape of the input. Fig. 14 illustrates a highly clustered non-manifold foliage model, which consists of numerous components. The general shape of the branches becomes noticeable in Fig. 14(d). Notice that throughout almost the whole refinement process, the foliage remains in the form of a single component. Fig. 8 depicts the refinement process of a human foot bone model. The original model consists of numerous components for different bones. These remain connected throughout almost the whole process and the approximation remains bounding the bones.

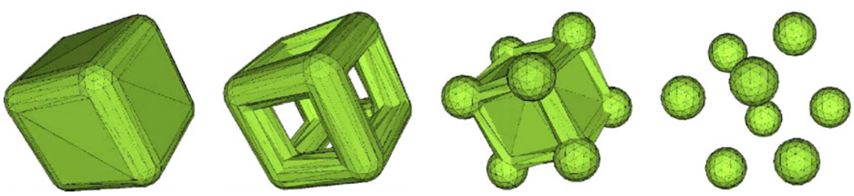
Table 1
The running times of the carving algorithm.

Model	Vertices	Hull vertices	Faces	Tetrahedra	Time (s)
Yucca leaves	801	87	1.2K	5.3K	0.3
Bones	2,154	124	4.2K	13.9K	1.3
Mushrooms	4,515	282	9.0K	30.0K	3.6
Elm	7,401	91	8.5K	47.4K	9.5
Horse	19,851	979	39.7K	137.9K	70.5
Scots Pine	90,125	151	61.7K	595.1K	847.9

The running time of our current implementation for the CDT carving algorithm is depicted in Table 1. The machine used to achieve the results has 500 MHz Intel Pentium processor and 496 MB of main memory. The running time of the CDT carving algorithm on each model consists of the running time required for constructing the constrained Delaunay tetrahedralization of the complement and for carving its tetrahedra. Our implementation makes use of the useful CDT library by Hang Si [28].

7. Conclusion and future work

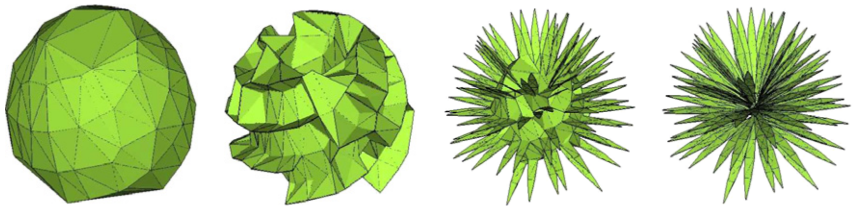
We have presented a refinement method for the simplification of polygonal surfaces. The method aims at producing a chain of approximations morphing to an input mesh of arbitrary topology from a genus-zero, coarse, bounding mesh. The morphing is achieved by manipulating the input's complement with respect to the bounding mesh. We have also described the CDT carving algorithm, which follows the guidelines of the method to construct a multiresolution stream of quality approximations. We have further described two modifications of the algorithm that allow controlling its effect and avoiding problems in resulting approximations. The algorithm is useful for simplification of highly clustered meshes, a class of meshes that tend to lose their basic characteristics and shape by any simplification method known to us so far. Viewing the approximation chain reversely as a decimation chain, the aggregation is monotonic, i.e., it constantly reduces the number of disjoint components. The approximations produced by the framework are silhouette-expanding, coarsened inside-out and converge to the convex hull.



(a) Convex hull. (b) $dp = 0$. (c) $dp = 10$. (d) Original model.

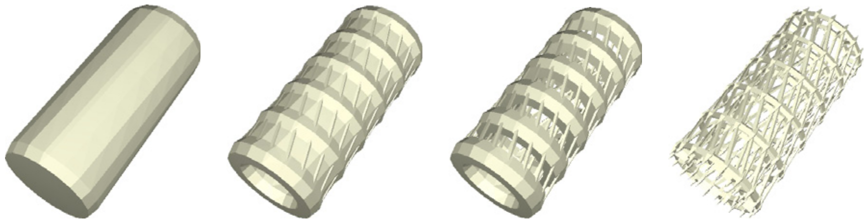
Fig. 10. Carving an eight-geosphere model with and without considering the depth of tetrahedra. (a) The convex hull of the model. (b) Result without depth parameterization. (c) Result with depth parameterization. (d) The input model.

Author's personal copy



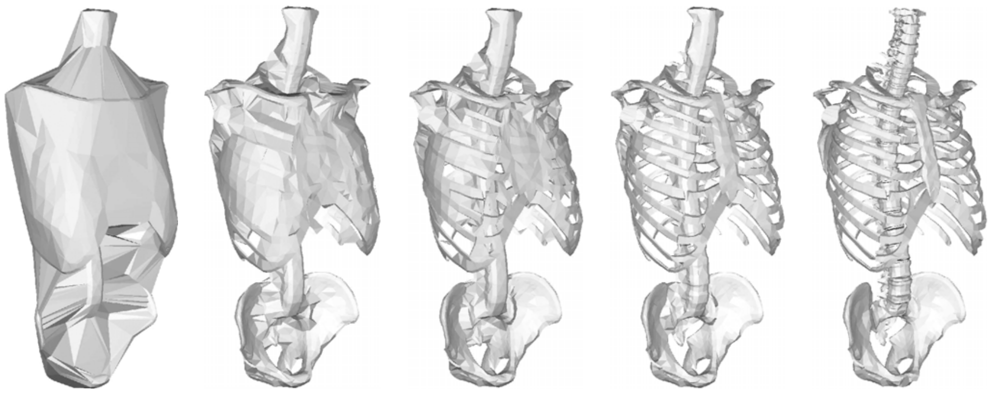
(a) Convex hull. (b) $bp = 0$. (c) $bp = 7.5$. (d) Original model.

Fig. 11. Carving a Yucca group of leaves model, with and without controlling the breadth of each step. (a) The convex hull. (b) Result without breadth parameterization. (c) Result with breadth parameterization. (d) The input model.



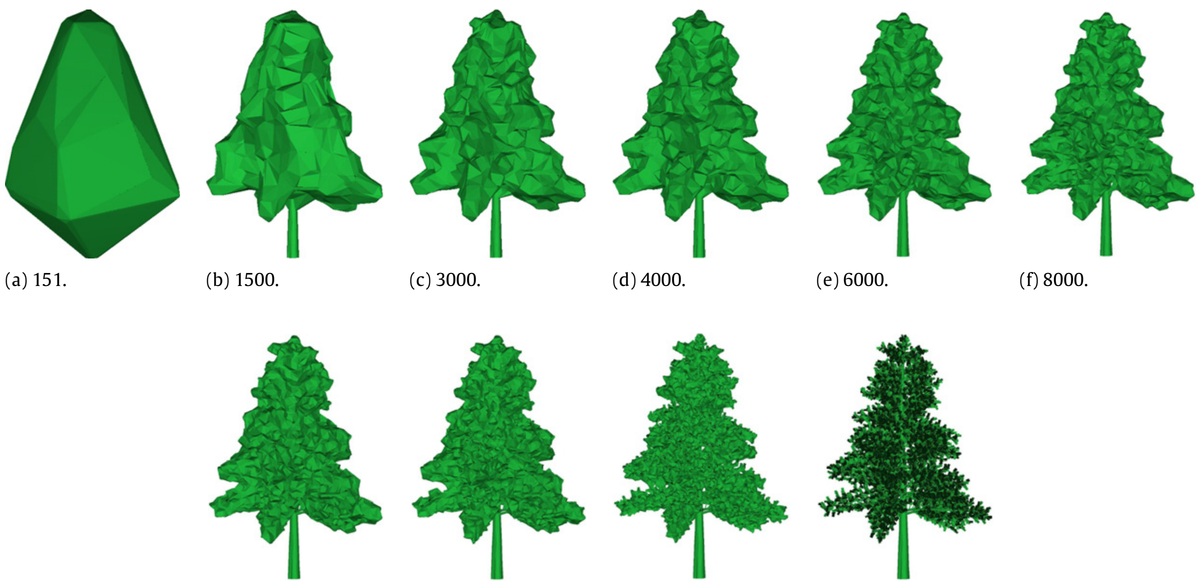
(a) 165 vertices. (b) 550 vertices. (c) 1350 vertices. (d) 4752 vertices.

Fig. 12. Topology-simplified levels of detail for a sconce model.



(a) 1400. (b) 4600. (c) 5500. (d) 7200. (e) 12,479.

Fig. 13. Simplified levels of detail for a torso model and their corresponding vertex counts.



(a) 151. (b) 1500. (c) 3000. (d) 4000. (e) 6000. (f) 8000.

(g) 10,000. (h) 14,000. (i) 33,000. (j) 90,125.

Fig. 14. Simplified levels of detail of a Scots Pine model and the number of vertices in each level.

Author's personal copy

The proposed approach has several limitations that should be addressed. As explained in Section 3.1, a lower boundary on the complexity of approximations is derived from the initial approximation used, e.g. the convex hull. In some cases the convex hull may include much of the complexity of the mesh. A possible solution is to first perform topology-preserving, non-degenerating geometric simplification on the input (e.g., edge-collapse). The convex hull of the simplified result can then be calculated and used as initial approximation for carving the simplified model. This yields a decimation stream in which the model is first coarsened geometrically and then topologically. One of the advantages of this approach is that the carving calculations are performed on a smaller dataset. Finally, our current CDT algorithm has relatively high time complexity. However, the carving process is performed offline in a preprocessing stage and generates approximations to be used online. The complexity of the algorithm is thus not critical.

This work can be extended in several directions. We would like to combine the algorithm with geometric simplification methods. This can be done by interleaving topology-preserving steps and carving steps. This will yield a general simplification method that correctly handles the topology. Another direction that can be explored is changing the vertex-placement strategy into an optimal placement scheme. We can select optimal locations for vertices in each step, rather than producing meshes whose vertex set is a subset of the original vertex set.

Acknowledgments

This work was supported by the Lynn and William Frankel Center for Computer Sciences. In addition, we would like to thank the reviewers for their constructive comments.

References

[1] Jimënez P, Thomas F, Torras C. 3d collision detection: A survey. Computers and Graphics 2001;25(2):269–85. URL citeseer.ist.psu.edu/431815.html.
[2] Garland M, Heckbert P. Surface simplification using quadric error metrics. In: Proceedings of SIGGRAPH'97, ACM SIGGRAPH. 1997. p. 209–16.
[3] Popović J, Hoppe H. Progressive simplicial complexes. In: Proceedings of SIGGRAPH'97, ACM SIGGRAPH. ACM Press; 1997. p. 217–24.
[4] Luebke D, Erikson C. View-dependent simplification of arbitrary polygonal environments. In: Proceedings of SIGGRAPH'97, ACM SIGGRAPH. ACM Press; 1997. p. 198–208.
[5] Rossignac J, Borrel P. Multi-resolution 3D approximations for rendering. In: Modeling in computer graphics. Springer-Verlag; 1993. p. 455–65.
[6] Edelsbrunner H. Weighted alpha shapes. Tech. rep. UIUCDCS-R-92-1760. Department of Computer Science, University of Illinois at Urbana-Champaign; 1992.
[7] Kobbelt LP, Vorsatz J, Labsik U, Seidel HP. A shrink wrapping approach to remeshing polygonal surfaces. Computer Graphics Forum 1999;18(3):119–30.
[8] Chew LP. Constrained Delaunay triangulations. Algorithmica 1989;4(1): 97–108.
[9] Shewchuk JR. Sweep algorithms for constructing higher-dimensional constrained Delaunay triangulations. In: Proceedings of the sixteenth annual symposium on computational geometry. 2000. p. 208–350.
[10] He T, Hong L, Varshney A, Wang S. Controlled topology simplification. IEEE Transactions on Visualization and Computer Graphics 1996;2(2):171–84.
[11] Wood Z, Hoppe H, Desbrun M, Schröder P. Removing excess topology from isosurfaces. ACM Transactions on Graphics 2004;23(2):190–208.
[12] Bischoff S, Kobbelt LP. Isosurface reconstruction with topology control. Pacific Graphics 2002;00:246. http://doi.ieeecomputersociety.org/10.1109/PCCGA.2002.1167868.
[13] Zhou Q-Y, Ju T, Hu S-M. Topology repair of solid models using skeletons. IEEE Transactions on Visualization and Computer Graphics 2007;13(4):675–85.
[14] Gerstner T, Pajarola R. Topology preserving and controlled topology simplifying multiresolution isosurface extraction. In: Proceedings of the IEEE visualization'00. 2000. p. 259–66.
[15] Andujar C, Ayala D, Brunet P, Joan-Arinyo R, Sole J. Automatic generation of multi-resolution boundary representations. Computer-Graphics Forum 1996; 15(3):87–96.
[16] Schroeder WJ. A topology modifying progressive decimation algorithm. In: IEEE visualization'97 proceedings. SIGGRAPH Press; 1997. p. 205–12.
[17] El-Sana J, Varshney A. Topology simplification for polygonal virtual environments. IEEE Transactions on Visualization and Computer Graphics 1998;4(2): 133–44.
[18] Ju T, Zhou Q-Y, Hu S-M. Editing the topology of 3d models by sketching. ACM Transactions on Graphics 2007;26(3):42. http://doi.acm.org/10.1145/1276377.1276430.
[19] Edelsbrunner H, Mücke E. Three-dimensional alpha shapes. ACM Transactions on Graphics 1994;13(1):43–72.
[20] Amenta N, Bern MW. Surface reconstruction by Voronoi filtering. In: Symposium on computational geometry. 1998. p. 39–48. URL citeseer.ist.psu.edu/amenta98surface.html.
[21] Bernardini F, Bajaj C.. Sampling and reconstructing manifolds using alpha-shapes. Tech. rep. CSD-97-013. Department of Computer Sciences, Purdue University; 1997.
[22] Attene M, Spagnuolo M. Automatic surface reconstruction from point sets in space. Computer Graphics Forum 2000;19(3):457–65.
[23] Gopi M, Krishnan S, Silva CT. Surface reconstruction based on lower dimensional localized Delaunay triangulation 19 (3). URL citeseer.ist.psu.edu/gopi00surface.html.
[24] Giesen J, John M. Surface reconstruction based on a dynamical system. Computer Graphics Forum 2003;21(3).
[25] Shewchuk JR. A condition guaranteeing the existence of higher-dimensional constrained Delaunay triangulations. In: Proceedings of the fourteenth annual symposium on computational geometry. Association for Computing Machinery; 1998. p. 76–85.
[26] Shewchuk JR. Delaunay refinement algorithms for triangular mesh generation. Computational Geometry: Theory and Applications 2002;22(1):86–95.
[27] Guéziec A, Taubin G, Lazarus F, Horn W. Converting sets of polygons to manifold surfaces by cutting and stitching. In: Proceedings of the IEEE visualization'98. 1998. p. 383–90.
[28] Si H. TetGen. A quality tetrahedral mesh generator and three-dimensional Delaunay triangulator. Version 1.4. User's manual. 2006. http://tetgen.berlios.de.