HHS Public Access

Author manuscript

ACM Trans Graph. Author manuscript; available in PMC 2020 August 20.

Published in final edited form as:

ACM Trans Graph. 2020 May ; 39(3): . doi:10.1145/3337680.

VoroCrust: Voronoi Meshing Without Clipping

AHMED ABDELKADER[*],
University of Maryland, College Park

CHANDRAJIT L. BAJAJ,
University of Texas, Austin

MOHAMED S. EBEIDA[†],
Sandia National Laboratories

AHMED H. MAHMOUD,
University of California, Davis

SCOTT A. MITCHELL,
Sandia National Laboratories

JOHN D. OWENS,
University of California, Davis

AHMAD A. RUSHDI
University of California, Davis and Sandia National Laboratories

Abstract

Polyhedral meshes are increasingly becoming an attractive option with particular advantages over traditional meshes for certain applications. What has been missing is a robust polyhedral meshing algorithm that can handle broad classes of domains exhibiting arbitrary curved boundaries and sharp features. In addition, the power of primal-dual mesh pairs, exemplified by Voronoi-Delaunay meshes, has been recognized as an important ingredient in numerous formulations. The VoroCrust algorithm is the first provably correct algorithm for conforming Voronoi meshing for non-convex and possibly non-manifold domains with guarantees on the quality of both surface and volume elements. A robust refinement process estimates a suitable sizing field that enables the careful placement of Voronoi seeds across the surface circumventing the need for clipping and avoiding its many drawbacks. The algorithm has the flexibility of filling the interior by either structured or random samples, while all sharp features are preserved in the output mesh. We demonstrate the capabilities of the algorithm on a variety of models and compare against state-of-the-art polyhedral meshing methods based on clipped Voronoi cells establishing the clear advantage of VoroCrust output.

[†]Correspondence address: msebeid@sandia.gov.
[*]Author names are listed in alphabetical order.

Keywords

Voronoi; Meshing; Refinement; Sharp Features; Union of Balls; Poisson-disk Sampling; Slivers

1 INTRODUCTION

The computational modeling of physical phenomena requires robust numerical algorithms and compatible high-quality domain discretizations. Finite element methods traditionally use simplicial meshes, where well-known angle conditions prohibit skinny elements [Shewchuk 2002]. The limited degrees of freedom of tetrahedral elements often lead to excessive refinement when modeling complex geometries or domains undergoing large deformations. This motivated generalizations to general polyhedral elements, which enjoy larger degrees of freedom and have recently been in increasing demand in computer graphics [Martin et al. 2008], physically-based simulations [Bishop 2014], applied mathematics [Manzini et al. 2014], computational mechanics [Gain et al. 2014a] and computational physics [Lipnikov et al. 2014].

While the generation of tetrahedral meshes based on Delaunay refinement [Cheng et al. 2012] or variational optimization [Alliez et al. 2005] is well established, research on polyhedral mesh generation is less mature. To further ensure the fidelity of the discrete model, the fundamental properties of continuum equations have to be preserved [Desbrun et al. 2008]. A well-principled framework is enabled through the combined use of primal meshes and their orthogonal duals [Mullen et al. 2011]. The power of orthogonal duals, exemplified by Voronoi-Delaunay meshes, has recently been demonstrated on a range of applications in computer graphics [Goes et al. 2014] and computational physics [Engwirda 2018]. It is therefore imperative to develop new algorithms for primal-dual polyhedral meshing.

In this paper, we present the design and implementation of VoroCrust: the first algorithm for meshing arbitrary nonconvex, non-smooth, and possibly non-manifold domains by conforming Voronoi meshes. The implicit output mesh, compactly encoded by a set of Voronoi seeds, comes with an orthogonal dual defined by the corresponding Delaunay tetrahedralization. This makes VoroCrust one of the first robust and efficient algorithms for primal-dual polyhedral meshing. The crux of the algorithm is a robust refinement process that estimates a suitable sizing function to guide the placement of Voronoi seeds. This enables VoroCrust to protect all sharp features, and mesh the surface and interior into quality elements. We demonstrate the performance of the algorithm through a variety of challenging models, see Figure 5, and compare against state-of-the-art polyhedral meshing methods based on clipped Voronoi cells; see Figures 1 and 2.

1.1 Background

Conventional mesh elements, as in tetrahedral and hexahedral meshes, often require excessive refinement when modeling complex geometries or domains undergoing large deformations, e.g., cutting, merging, fracturing, or adaptive refinement [Chen et al. 2014; Clausen et al. 2013; Wicke et al. 2010; Wojtan et al. 2009]. A key advantage of general

polyhedral elements is their superior ability to adjust to deformation [Gain et al. 2014b; Martin et al. 2008] and topological changes [Wu et al. 2015], while being less biased to principal directions compared to regular tessellations [Talischi et al. 2013]. In addition, polyhedral elements typically have more neighbors, even at corners and boundaries, enabling better approximation of gradients and possibly higher accuracy using the same number of conventional elements [Peric and Ferguson 2005].

Unfortunately, robust polyhedral meshing algorithms are still lacking. State-of-the-art approaches often rely on *clipping*, i.e., truncating cells of an initial mesh to fit the domain boundaries [Yan et al. 2010]. Such an initial mesh can be obtained as a Voronoi mesh, e.g., with seeds randomly generated inside the domain [Ebeida and Mitchell 2012] or optimized by centroidal Voronoi tessellations (CVT) [Yan et al. 2010], possibly taking anisotropy into account [Budninskiy et al. 2016]. Alternatively, an initial Voronoi mesh can be obtained by dualizing a conforming tetrahedral mesh [Garimella et al. 2014]. Although no clipping is needed if the tetrahedralization is *well-centered*, generating such meshes is very challenging and only heuristic solutions are known [VanderZee et al. 2010]. A weaker *Gabriel property* ensures all tetrahedra have circumcenters inside the domain and can be guaranteed for polyhedral domains with bounded minimum angles [Si et al. 2010]; however, the dual Voronoi cells still need to be clipped.

While clipping can be implemented efficiently, it fails to produce true Voronoi cells, sacrificing key geometric properties [Ebeida and Mitchell 2012]. Specifically, clipping at sharp features may yield cells that are not convex as shown in Figure 2. This violates the requirements of several important applications, e.g., barycentric interpolation [Warren et al. 2007] and polyhedral finite elements [Wicke et al. 2007]. Even more general formulations, like Virtual Element Methods [Beirão da Veiga et al. 2013], require star-shaped cells which clipping cannot guarantee. Unlike prior work, VoroCrust robustly meshes complex domains into Voronoi cells that naturally conform to the boundary circumventing the need for clipping. The combined use of such Voronoi meshes and their orthogonal duals, defined by the corresponding Delaunay tetrahedralizations [Aurenhammer et al. 2013], has been recognized as an important framework for computational modeling [Desbrun et al. 2008; Mullen et al. 2011].

1.2 Related Work

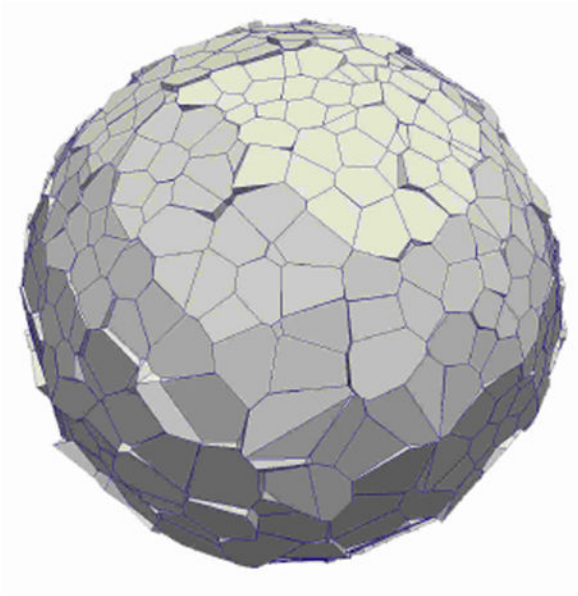
We further motivate Voronoi meshing through a detailed review of primal-dual meshing and its practical relevance. Then, we proceed to review related work on Voronoi-based modeling and meshing piecewise-smooth complexes, which together provide the theoretical underpinnings of the VoroCrust algorithm.

Orthogonal Primal-Dual Meshing.—Orthogonal primal-dual mesh pairs are unstructured staggered meshes [Harlow and Welch 1965] with desirable conservation properties [Perot 2000], enabling discretizations that closely mimic the continuum equations being modeled [Bochev and Hyman 2006; Desbrun et al. 2008]. The power of orthogonal duals [Mullen et al. 2011] was recognized in early works on structural design [Maxwell 1870; Rankine 1864] and numerical methods [Macneal 1953], and has recently been

demonstrated on a range of applications in computer graphics [Goes et al. 2014], self-supporting structures [Akbarzadeh et al. 2015], mesh parameterization [Mercat 2001], and computational physics [Engwirda 2018]. In particular, Voronoi-Delaunay meshes are the default geometric realization of many formulations in numerical methods [Nicolaides and Wu 1997], fluid animation [Elcott et al. 2007], fracture modeling [Sukumar and Bolander 2009], and computational cell biology [Novak et al. 2007].

Despite many attempts to design a robust Voronoi meshing algorithm, a general solution to the problem remained elusive. In particular, a number of widely used numerical simulators for flow and transport models, e.g., TOUGH2 [Pruess 1991] and PFLOTRAN [Lichtner et al. 2015], compute gradients along nodal lines connecting neighboring cells, and hence require that these dual edges are orthogonal to the common primal facet [Pruess 2004]. Several heuristic approaches to the generation of Voronoi meshes for such simulators were developed [Bonduà et al. 2017; Freeman et al. 2014; Hu et al. 2016; Kim et al. 2015; S. Klemetsdal et al. 2017]. The situation is further complicated for multi-material domains, where the difficulty of generating conforming meshes necessitates dealing with mixed elements straddling the interface between multiple materials [Dawes 2017; Garimella and Lipnikov 2011; Kikinzon et al. 2017]. In contrast, VoroCrust is a well-principled algorithm for conforming Voronoi meshing that can handle a large class of domains having as boundary a possibly nonmanifold surface with arbitrarily sharp features.

Voronoi-based Modeling.—



An intuitive approach to surface reconstruction is to place pairs of Voronoi seeds *mirrored* across the surface such that their shared Voronoi facets approximate the surface; see Figure 3(a). However, a naive implementation of this idea results in a rough surface with spurious misaligned facets; see the inset and Figure 3(b). One such mirroring approach relies on an input sizing parameter to segment images into convex polygons assuming no four lines meet in a point [Duan and Lafarge 2015].

Nonetheless, a more principled mirroring approach provided the first provably-correct surface reconstruction algorithm [Amenta and Bern 1999]. Given an ϵ-sample from an unknown smooth surface, the PowerCrust algorithm [Amenta et al. 2001] places weighted Voronoi seeds at a subset of the vertices in the Voronoi diagram of the input samples. While PowerCrust successfully avoids misaligned facets, the placement of seeds as described is restricted to lie close to the medial axis resulting in very skinny Voronoi cells extending perpendicularly to the surface; see Figure 3(c). For the purposes of conforming 3D Voronoi meshing, it is necessary to avoid such skinny cells. In contrast, VoroCrust is able to capture the surface using pairs of unweighted seeds placed close to the surface, enabling further decomposition of the interior using additional seeds; see Figure 3(d). A visual summary of the VoroCrust algorithm is provided in Figure 4.

An abstract version of the VoroCrust algorithm for smooth manifold surfaces was recently analyzed by Abdelkader et al. [2018]. Assuming access to the local feature size with an ϵ-sampling given as input, strong theoretical guarantees on the output quality were established [Abdelkader et al. 2018]. In contrast, this paper describes a practical realization of the VoroCrust algorithm for domains with non-manifold boundaries exhibiting arbitrarily sharp features and narrow regions. The VoroCrust refinement produces a union of balls that protects all sharp features while satisfying similar properties to the one analyzed by Abdelkader et al. [2018]. Hence, we retain all the approximation and quality guarantees they established, except in the vicinity of sharp features where quality bounds necessarily deteriorate to conform to those features. Furthermore, certain ball configurations yield undesirable sliver artifacts in the output surface and their elimination was left as future work [Abdelkader et al. 2018]. The proposed VoroCrust algorithm provably eliminates all such slivers.

The simpler related problem of generating a Voronoi mesh that conforms exactly to restricted classes of piecewise-linear complexes was studied earlier by Abdelkader et al. [2017a]. The approach they adopted uses simple rules for the placement of Voronoi seeds to reproduce an input piecewise-linear complex as a set of Voronoi faces similar to earlier works on conforming Delaunay meshing [Cohen-Steiner et al. 2002; Murphy et al. 2001; Rand and Walkington 2009]. In contrast, VoroCrust always retains the topology of the domain but is not restricted to conform exactly to the boundary; it effectively performs remeshing to improve the output quality within the tolerance specified by the input parameters.

Meshing Piecewise-smooth Complexes.—Delaunay refinement (DR) is a very successful algorithm for the generation of quality unstructured tetrahedral meshes [Cheng et al. 2012]. Since the presence of small angles in the input domain may threaten the termination of DR, a lower bound on input angles may be necessary. Following a series of works extending DR to more general classes of domains, e.g., polyhedral domains with arbitrarily small angles [Cheng and Poon 2006] and domains with curved boundaries assuming lower bounds on the smallest angles [Oudot et al. 2010; Tournois et al. 2009], Cheng et al. [2010] were finally able to eliminate the angle constraints for a large class of inputs called piecewise-smooth complexes. They achieved that by deriving a feature size that blends the definitions used for smooth and polyhedral domains, ensuring the protection of

sharp features. However, their algorithm is largely impractical as it relies on expensive predicates evaluated using the equations of the underlying surface. To obtain a practical variant, Dey and Levin [2009] relied on an input threshold to guide refinement, where topological correctness can only be guaranteed if it is sufficiently small. Another issue with using such a threshold is the uniform sizing of the output mesh, since adaptive sizing requires better sensitivity to the underlying surface. In contrast, the proposed VoroCrust refinement leverages the quality of the input mesh to automatically estimate a sizing similar to the one defined by Cheng et al. [2010]; this enables VoroCrust to retain the superior guarantees they established while being practical and efficient as shown in our results.

1.3 Contributions

The VoroCrust algorithm is the first algorithm for conforming Voronoi meshing that can handle a large class of domains with both curved boundaries and arbitrarily sharp features. VoroCrust circumvents the need for clipping, which is the current standard for polyhedral Voronoi-based meshing, successfully avoiding its drawbacks. VoroCrust has the flexibility of decomposing the interior into convex Voronoi cells using either structured or randomly generated seeds. The resulting seeds compactly and uniquely encode the Voronoi mesh, which can be explicitly constructed on-the-fly in a local fashion.

In a broader sense, VoroCrust is one of the first robust and efficient algorithms for polyhedral meshing. In particular, the VoroCrust output consisting of true unweighted Voronoi cells decomposes the domain into convex cells and comes with an orthogonal dual Delaunay tetrahedralization. Such convex decompositions and primal-dual mesh pairs are very useful, and sometimes necessary, in many applications.

The crux of the algorithm is a robust and well-principled refinement process that converges to a suitable sizing function enabling the placement of Voronoi seeds to approximate the surface while preserving all sharp features. VoroCrust estimates sizing through refinement [Rand and Walkington 2008] as applied in modern meshing frameworks [Tournois et al. 2009]. This paradigm has proven more efficient than the more traditional approach based on medial axis approximations, e.g., Alliez et al. [2005]. The advantage of VoroCrust output is demonstrated by an extensive comparison against state-of-the-art polyhedral meshing methods based on clipped Voronoi cells [Yan et al. 2010]; see Figures 1 and 2.

The practicality of the proposed VoroCrust algorithm and the quality of its output further stem from additional design ingredients to speed up various computations while satisfying the requirements on sampling. We demonstrate the performance of the algorithm through a variety of challenging models, see Figure 5, and include a comprehensive parameter study to test the algorithm at the limits.

2 THE VOROCRUST ALGORITHM

Given a representation of a domain $\mathcal{O}$, the algorithm produces a boundary-conforming Voronoi decomposition. The crux of the algorithm is the generation of a set of weighted surface samples corresponding to a set of balls $\mathcal{B}$ whose union $\mathcal{U} = \cup \mathcal{B}$ approximates the boundary $\mathcal{M} = \partial\mathcal{O}$. Specifically, $\mathcal{U}$ covers $\mathcal{M}$ and has the same topology. In addition, $\mathcal{U}$

captures the sharp features of $\mathscr{M}$. To further guarantee the quality of surface approximation, the radii of surface balls vary smoothly and are sufficiently small w.r.t. the local curvature of $\mathscr{M}$. In other words, the radii of balls in $\mathscr{B}$ mimic a local feature size for $\mathscr{M}$. Finally, certain configurations of balls are perturbed to eliminate undesirable artifacts in the output surface mesh. These requirements are used to design a refinement process that converges to a suitable union of balls. The conforming surface mesh is obtained by essentially dualizing $\mathscr{U}$ to obtain a set of Voronoi seeds $\mathcal{S}^{\updownarrow}$. Once $\mathscr{U}$ is obtained, the interior is easily meshed by sampling additional seeds $\mathcal{S}^{\downarrow\downarrow}$ outside $\mathscr{U}$. The output mesh can then be computed as a subset of the Voronoi diagram of the seeds in $\mathcal{S}^{\updownarrow} \cup \mathcal{S}^{\downarrow\downarrow}$ without any clipping. In the remainder of this section, we elaborate on these steps per the high-level pseudocode in Algorithm 1 and Figure 4.

2.1 Input

VoroCrust can handle a domain $\mathscr{O}$ having as boundary a possibly non-manifold piecewise-smooth complex (PSC) $\mathscr{M}$. The boundary PSC $\mathscr{M}$ possibly contains *sharp features* where the normal to the surface does not vary smoothly. We make no assumption on how small the input angles might be at such sharp features. VoroCrust guarantees the preservation of all sharp features; sharp corners appear exactly as vertices, while sharp creases are approximated by a set of edges.

Input Mesh.—The algorithm takes as input a watertight piecewise-linear complex (PLC) $\mathscr{T}$ approximating the boundary $\mathscr{M}$. As in [Dey and Ray 2010], we assume that $\mathscr{T}$ approximates $\mathscr{M}$ in terms of both the Hausdorff error and the surface normals; this enables various predicates to be evaluated using the input PLC rather than the equations describing the underlying PSC [Cheng et al. 2010]. In particular, we assume that all dihedral angles in the input mesh, except at sharp features, are at least $\pi - \theta^{\natural}$, where the *smoothness threshold* $\theta^{\natural} > 0$ is an implicit design parameter. For the current implementation, we assume $\mathscr{T}$ is a triangle mesh with no self-intersection. Well-established methods can be used to obtain such a mesh given a suitable representation of the domain $\mathscr{O}$ [Dey and Levine 2009; Hu et al. 2018; Tournois et al. 2009].

Parameters.—The algorithm also takes the following inputs:

- *sz*: a sizing field which indicates the desired *size* of mesh elements, and defaults to the diameter of $\mathscr{T}$ or ∞.
- $\theta^{\#} < \frac{\pi}{2}$: an angle threshold used to identify the *sharp features* in the PLC $\mathscr{T}$ and bound approximation errors.
- $L < 1$: a *Lipschitz* parameter that bounds the variation of radii in $\mathscr{B}$ and helps speed-up proximity queries.

Algorithm 1:

High-level VoroCrust algorithm

Input: PLC $\mathcal{T}$ approximating the domain $\mathcal{O}$, sizing field sz, and parameters $\theta^\sharp$ and L (Section 2.1)

$\mathcal{F} \leftarrow$ the set of sharp features w.r.t. $\theta^\sharp$ (Section 2.2)

$\mathcal{B} \leftarrow$ a set of balls protecting all features in $\mathcal{F}$ (Section 2.3)

while $\mathcal{U} = \cup \mathcal{B}$ *does not cover* $\mathcal{T}$ **do**

  Add balls to recover the protection of $\mathcal{F}$ and cover $\mathcal{T}$

  Shrink balls violating any ball conditions (Section 2.3)

  Shrink balls or forming half-covered seeds (Section 2.4)

end

$\mathcal{S}^{\updownarrow} \leftarrow$ pairs of seeds from triplets of balls in $\mathcal{B}$ (Section 2.4)

$\mathcal{S}^{\downarrow\downarrow} \leftarrow$ seeds sampled from the interior of $\mathcal{O}\backslash\mathcal{U}$ (Section 2.5)

return $\mathcal{S}^{\updownarrow} \cup \mathcal{S}^{\downarrow\downarrow}$

2.2 Preprocessing

Before refinement, VoroCrust indexes the elements of the input PLC $\mathcal{T}$ and enforces the smoothness condition per the parameter $\theta^\flat$. Then, the algorithm constructs a number of data structures for proximity queries against $\mathcal{T}$ and $\mathcal{B}$.

Feature Detection.—We define a *sharp edge* as an edge of $\mathcal{T}$ subtending a dihedral angle less than $\pi - \theta^\sharp$, or any nonmanifold edge incident to exactly one or more than two facets. These sharp edges partition the set of facets incident to any fixed vertex into *sectors*. We define a *sharp corner* as a vertex of $\mathcal{T}$ incident to more than two sharp edges, or two sharp edges whose supporting lines make an angle less than $\pi - \theta^\sharp$, or two facets in the same sector whose normals differ by at least $\theta^\sharp$. A polyline arising from a chain of connected sharp edges is called a *crease*, and either forms a cycle or connects two sharp corners. The connected components of the boundary containing no sharp features, denoted $\mathcal{T}_S$, are called *surface patches*. The collection of sharp corners, creases and surface patches are collectively referred to as the *strata* of $\mathcal{T}$.

The algorithm uses $\theta^\sharp$ to test each edge in $\mathcal{T}$, and collects all sharp edges in a set E. Then, each vertex is tested using $\theta^\sharp$ and E, and the sharp corners are collected into the set $\mathcal{F}_C$. From E and $\mathcal{F}_C$, connected chains of sharp edges are collected into the set $\mathcal{F}_E$ by flooding through common vertices except for sharp corners. As a byproduct, each crease is given an index and an orientation, applied consistently to all its sharp edges. Similarly, the facets of $\mathcal{T}$ are indexed, oriented and collected into the set of surface patches $\mathcal{T}_S$ by flooding across non-sharp edges. Finally, we set $\mathcal{F} = \mathcal{F}_C \cup \mathcal{F}_E$.

Patch Smoothing.—If the input mesh $\mathcal{T}$ does not satisfy the required bound on dihedral angles in terms of $\theta^\flat$, VoroCrust starts by applying adaptive loop subdivision [Amresh et al.

2003] to ensure all dihedral angles subtended by neighboring facets in the same surface patch in $\mathcal{T}_S$ are sufficiently large. In our implementation, we run 6 iterations of loop subdivision, applying subdivision adaptively such that facets subtending dihedral angles larger than 175° are not subdivided. Typical values of $\theta^\flat$ resulting from this step range from 10°to 15°.

Proximity Queries.—Upon generating a new sample point $p \in \mathcal{T}$, VoroCrust needs to find the balls in $\mathcal{B}$ covering p, and estimate its distance to the elements of $\mathcal{T}$ satisfying certain conditions w.r.t. $\theta^\sharp$. To speed up such queries, the algorithm constructs three *boundary k-d trees* to index the elements in $\mathcal{F}_C$, $\mathcal{F}_E$ and $\mathcal{T}_S$. The k-d trees for $\mathcal{F}_E$ and $\mathcal{T}_S$ are populated by supersampling the respective elements with a large number of samples proportional to their sizes. Similarly, the balls in $\mathcal{B}$ are indexed into three *ball k-d trees*. When querying the ball k-d trees for balls in the neighborhood of a given point, the L-Lipschitzness of ball radii helps to bound the range and overhead of such queries; see the appendix for more details.

2.3 Ball Refinement

At a high level, the desired union of balls $\mathcal{U}$ has to (1) protect the sharp features of $\mathcal{T}$ as in [Cheng et al. 2010], and (2) cover $\mathcal{T}$ while matching its topology as in [Abdelkader et al. 2018]. VoroCrust achieves this through a set of *ball conditions* imposed on the balls in $\mathcal{B}$. Violations of these conditions drive a refinement process which converges to a suitable union of balls. Before describing this process, we introduce a number of definitions and subroutines.

Smooth Neighborhoods.—As in [Cheng et al. 2010], we appeal to the curvature of the surface to infer a suitable notion of sizing. Fix a point $x \in \mathcal{T}$ and let σ be a face of $\mathcal{T}$ containing x. If σ is a sharp edge, define $v_{x,\sigma}$ as a unit vector parallel to σ. If σ is a surface patch, define $v_{x,\sigma}$ as a unit vector normal to σ. $v_{x,\sigma}$ inherits the orientation of the stratum, i.e., crease or surface patch, containing σ. A path γ lying entirely in a unique stratum Σ is called a *smooth path* iff for all $x,y \in \gamma$ we have that $\angle v_{x,\sigma}, v_{y,\tau} \leq \theta^\sharp$, where σ and τ are the two top-dimensional faces of Σ containing x and y, respectively. Two points $x,y \in \mathcal{T}$ are called *co-smooth* iff they can be connected by a smooth path.

Ball Conditions.—For a sample point $p \in \mathcal{T}$, let $b_p \in \mathcal{B}$ denote the ball centered at p and let r_p denote its radius. The following conditions drive the refinement process and are ensured for $\mathcal{B}$ upon termination; see Figure 6.

(C1) Smooth Coverage.: For any $b_p \in \mathcal{B}$ and all $x \in b_p \cap \mathcal{T}$, we require that p and x are co-smooth.

(C2) Smooth Overlaps.: For any $b_p, b_q \in \mathcal{B}$ s.t. $b_p \cap b_q = \phi$, we require that $b_p \cup b_q$ contains a smooth path from p to q.

(C3) Local L-Lipschitzness.: For any two balls b_p, $b_q \in \mathcal{B}$ such that $p, q \in \mathcal{F}_C$, or $p, q \in \mathcal{F}_E$, or $p, q \in \mathcal{T}_S$, we require that $r_p \leq r_q + L \cdot \|p - q\|$.

(C4) Deep Coverage.: Fix a constant $\alpha \in (0, 1)$. For all $x \in \mathcal{T}$, we require that $\|x - p\| \leq (1 - \alpha) \cdot r_p$ for some ball $b_p \in \mathcal{B}$. In addition, we require that $\|p - q\| \geq (1 - \alpha) \cdot \max(r_p, r_q)$ for all balls $b_p, b_q \in \mathcal{B}$.

Sizing Estimation.—A *sizing* assigns to each new sample p a radius r_p. We seek a sizing at most sz that satisfies all ball conditions. VoroCrust computes such a sizing by dynamically evolving the assignments r_p for each ball $b_p \in \mathcal{B}$ in the course of the refinement process. To speed up convergence, a newly generated ball b_p is initialized with a conservative estimate that is more likely to satisfy all ball conditions. To help avoid C1 and C2 violations, the boundary k-d trees are queried using p to obtain a surrogate point q^* for the nearest non-cosmooth point on $\mathcal{T}$. To help avoid C3 violations, the ball k-d trees are queried to find the ball b_q whose center is nearest to p. With that, we set $r_p = \min(sz(p), 0.49 \cdot \|p - q^*\|, r_q + L \cdot \|p - q\|)$.

Termination.—Since VoroCrust uses the PLC $\mathcal{T}$, which only provides a discrete approximation to the PSC $\mathcal{M}$, and approximates various distance queries, the sizing estimates as defined above may later be found to violate some ball conditions. By similar arguments to those in [Dey and Levine 2009], refinement terminates satisfying all ball conditions. The intuition is that for each region on a crease or surface patch, there exists a positive lower bound on ball radii below which neither of the first two conditions can be violated. The refinement process resolves violations by *shrinking* some balls, effectively adjusting all sizing estimates, before recursing to restore protection and coverage. As demonstrated through a variety of challenging models, our algorithm is tuned to avoid excessive refinement; see Section 3.

Sampling Basics.—The refinement process uses Maximal Poisson-Disk Sampling (MPS) [Ebeida et al. 2012; Guo et al. 2015; Yan and Wonka 2013] to generate the balls needed to protect the creases and cover the surface patches. The MPS procedure maintains an *active pool*, initialized by all faces on the stratum at hand. To generate a new sample, MPS starts by sampling a face σ from the active pool with a probability proportional to its measure, defined as the length for edges and the area for facets. Then, a point p is sampled from σ uniformly at random. If p is not covered by the balls in $\mathcal{B}$, it is assigned a radius r_p and the ball b_p is added into $\mathcal{B}$. Otherwise, p is discarded and a *miss counter* is incremented. Upon counting 100 successive misses, all faces in the active pool are *subdivided* into *subfaces* and the miss counter is reset; edges are split in half and facets are evenly split into four by connecting edge midpoints. Any subface whose points are all deeply covered is discarded, and the remaining subfaces become the new active pool.

Deep Coverage. For any point $x \in \mathcal{T}$, condition C4 dictates a stronger form of coverage by the balls in $\mathcal{B}$. We say that $x \in \mathcal{T}$ is α*-deeply covered* by a ball $bp \in \mathcal{B}$ if $\|p - x\| \leq (1 - \alpha) \cdot r_p$; see Figure 6. We set $\alpha = 1 - \frac{\sqrt{3}}{2} \approx 0.13$ in our implementation.

Equivalently, we require adjacent balls to intersect *deeply*. The reason for that is twofold. First, any point x in the proximity of a crease Σ must be closer to the weighted samples on Σ than the samples on any other stratum of $\mathcal{T}$ [Dey and Levine 2009]. Second, a sufficient

distance between pairs of seeds is needed to bound the aspect ratio of Voronoi cells [Abdelkader et al. 2018]. The refinement process ensures C4 by modifying the coverage test for MPS as follows. First, a new sample is only accepted if it is *not* deeply covered. Second, upon subdividing a face in the active pool, a subface is discarded only if it is completely deeply covered by a single ball with a co-smooth center. Third, the requirements of protecting sharp features prohibit deep overlaps between balls of different types; we elaborate on this further below following the description of our MPS implementation.

Detecting Violations.—Before MPS discards a subface σ, the algorithm checks for violations of C1 or C2, and shrinks encroaching balls as follows. The algorithm starts by finding the nearest sample to σ on each stratum using the respective ball k-d tree. Then, the algorithm queries the trees for neighboring balls and checks whether σ is deeply covered by any of these balls. For each such ball b_p, the algorithm also checks whether p is co-smooth with the points of σ. If not, the algorithm finds the point $q^* \in \sigma$ minimizing the distance to p and shrinks b_p if necessary to ensure $r_p \leq 0.49 \cdot \|p - q^*\|$. By ensuring such b_p does not overlap σ, C1 violations are avoided. In addition, letting τ denote the subface containing p, any ball b_q with $q \in \sigma$ cannot overlap b_p. This effectively avoids C2 violations as the algorithm ensures $\max(r_p, r_q) \leq 0.49 \cdot \|p - q^*\|$ before σ and τ are both discarded. Finally, whenever the algorithm shrinks a ball, it needs to check for violations of C3 and possibly shrink more balls; the algorithm in [Tournois et al. 2009] is similar in that regard. However, violations of C3 are not checked during the MPS procedure, which possibly terminates with such violations. As we describe below, enforcing C3 is interleaved with a later step to speed up convergence.

Testing Co-smoothness.—Given two subfaces σ, τ on a stratum Σ and a point $p \in \tau$, our implementation uses a more practical test rather than computing smooth paths on Σ. This test is based on the observation that smooth paths starting at a subface σ are confined to small (co)cones of aperture $2\theta^\#$ emanating from the boundary of σ. In particular, the smooth neighborhood is nearly collinear or coplanar with σ if Σ is a crease or surface patch, respectively.

The algorithm starts by finding the point $q^* \in \sigma$ minimizing the distance to p, and sets $v_{pq^*} = p - q^*$. Then, the co-smoothness test is relaxed to only require that (1) $\angle v_{\sigma, q^*}, v_{\tau, p} \leq \theta^\#$ and (2) $\angle v_{\sigma, q^*}, v_{pq^*} \leq \theta^\#$ if Σ is a crease, or $\angle v_{\sigma, q^*}, v_{pq^*} \leq \frac{\pi}{2} - \theta^\#$ if Σ is a surface patch. We argue that this relaxed test suffices for the refinement process to eventually guarantee both C1 and C2. Let $\gamma \in \Sigma$ be any path from p to σ. If γ is a smooth path, then the test passes on all subfaces along γ. Otherwise, the test fails for some subface $\sigma' \in \gamma$. Hence, if no smooth path exists from p to σ, then every such path γ encounters a subface σ' for which the test fails before reaching σ. By applying the relaxed test to every subface σ and each ball in a sufficiently large neighborhood around σ, any remaining violations of C1 or C2 can be detected before MPS terminates. To further validate this claim, we implemented the strict test and verified that both C1 and C2 are always satisfied when MPS terminates.

Protection and Coverage.—The refinement process is realized as a recursive MPS procedure (RMPS) that goes through three phases, ordered by the dimension of the underlying stratum, starting with the protection of sharp corners to the protection of creases and finally the coverage of surface patches; see Figure 7. At each phase, if refinement shrinks any of the balls belonging to a previous phase, the algorithm recurses by rerunning RMPS on the affected lower-dimensional strata before proceeding. The process starts by initializing the set of balls with one *corner ball* centered at each sharp corner. As the base case of RMPS, the algorithm enforces C3 among corner balls by brute force, i.e., each ball is checked against the rest and is shrunk as needed. Then, each crease Σ is protected by a set of *edge balls* by running RMPS on Σ. If any corner ball had to be shrunk, RMPS immediately recurses to adjust the corner balls. Whenever RMPS terminates on all creases, the algorithm enforces C3 on all edge balls and reruns RMPS as needed to restore protection. After successfully protecting all sharp corners and creases, the algorithm proceeds to cover each surface patch Σ by a set of *surface balls* by running RMPS on Σ. Similarly, if any corner or edge ball had to be shrunk, RMPS immediately recurses to the respective phase. Finally, the algorithm enforces C3 on surface balls. Before rerunning RMPS as needed to restore protection and coverage, the algorithm perturbs slivers, as we describe in Section 2.4; this helps refinement converge in fewer iterations.

We now turn back to the restrictions on overlaps between balls of different type. Whenever a subface encountered by RMPS is completely contained in a corner ball, it is excluded from RMPS in higher phases on neighboring strata. Similarly, whenever a subface is completely contained in an edge ball, it is excluded from RMPS on neighboring surface patches. This is necessary to ensure the protection of sharp features. As a consequence, the deep coverage condition C4 may be violated in the vicinity of sharp features. This contributes to the deterioration of element quality in these neighborhoods but otherwise does not threaten the termination of the algorithm; see Section 2.4 and the supplemental materials.

Density Regulation.—Extra care is needed to avoid the wellknown clustering phenomenon resulting from the greedy generation of samples. This can be mitigated by biasing the sampling to avoid introducing new sample points near the boundaries of existing balls. In particular, whenever the radius assigned to a new sample p results in the ball b_p violating C4 by containing an existing sample, p is rejected with a small constant probability; we set this constant to 0.1 in our implementation. If p is not rejected, b_p is shrunk to ensure it satisfies C4. As demonstrated in Section 3, VoroCrust successfully avoids unnecessarily dense clusters of samples.

2.4 Surface Meshing

VoroCrust populates the set of *surface seeds* $\mathcal{S}^{\updownarrow}$ using triplets of overlapping balls in $\mathscr{B}$. The bounding spheres of each such triplet intersect in exactly two points on either side of the boundary. The algorithm places one labeled Voronoi seed at each such point as long as it does not lie in the interior of any fourth ball in $\mathscr{B}$. Then, the Voronoi facets common to two Voronoi seeds on different sides of the boundary constitute the resulting VoroCrust surface mesh which coincides with the medial axis of the union of balls $\mathscr{U}$ [Amenta and Kolluri 2001], inheriting the topology of $\mathcal{T}$. The deep coverage condition C4 guarantees that all

samples p appear as vertices in the Voronoi diagram of $\mathcal{S}^{\updownarrow}$, with at least 4 seeds lying on ∂b_p. We point out that VoroCrust effectively remeshes the surface on-the-fly to reduce the complexity of the output within the tolerance specified by the input parameters. The quality of surface elements follows from L-Lipschitzness [Abdelkader et al. 2018], with the exception of elements formed by corner or edge balls in the vicinity of sharp features.

Sliver Elimination.—VoroCrust applies further refinement to the set of balls $\mathcal{B}$ to eliminate undesirable artifacts in the output. When a triplet of overlapping balls yield only one Voronoi seed, we have a *half-covered seed pair*. The four samples yielding the problematic configuration of balls are the vertices of a nearly flat tetrahedron in the weighted Delaunay triangulation defined by the samples [Abdelkader et al. 2018], hence we refer to them as *slivers*. These slivers result in extra *Steiner vertices*, besides the samples, appearing in the Voronoi diagram of the seeds and consequently on the output surface mesh. As these Steiner vertices may not lie on the input surface, their incident Voronoi facets may not be aligned with the surface possibly yielding large deviations in surface normals; see Figure 8. To eliminate such slivers, the algorithm determines one ball to shrink for each half-covered seed.

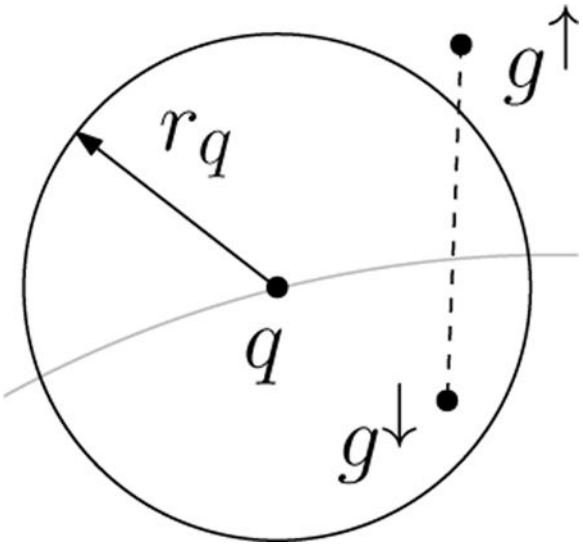
For every ball $b_p \in \mathcal{B}$, the algorithm queries the ball k-d trees for neighboring balls and collects those overlapping b_p into the set $\mathcal{B}_p$. The algorithm iterates over $\mathcal{B}_p$ to form triplets of overlapping balls including b_p. For each such triplet t, the algorithm computes the pair of intersection points on their bounding spheres and tests whether the pair is half-covered by any fourth ball in $\mathcal{B}_p$; all candidate fourth balls along with the triplet in t are collected into a secondary set $\mathcal{B}_t$. Then, every quartet of balls in $\binom{\mathcal{B}_t}{4}$ defining a half-covered seed pair is considered in isolation. For each such quartet, the algorithm determines the ball requiring the least shrinkage to uncover all seeds. Over all quartets in $\binom{\mathcal{B}_t}{4}$, the ball requiring the least shrinkage is assigned a smaller radius. For each ball b, the algorithm records the smallest radius assigned to b over all quartets it is part of. Once all balls are processed, the algorithm shrinks every ball assigned a smaller radius. Recalling that L-Lipschitzness is satisfied for $\mathcal{B}, |\mathcal{B}_p|$ is kept small and the running time of this procedure is linear in $|\mathcal{B}|$. The procedure just described eliminates a subset of existing slivers but potentially violates some ball conditions and creates new slivers. The algorithm reruns RMPS to resolve such violations before repeating to eliminate any remaining slivers.

Termination without Slivers.—Each execution of the above procedure, followed by rerunning RMPS, counts as a single iteration of sliver elimination. The termination of the algorithm requires a finite bound on the number of such iterations, which can be established by bounding the shrinkage that may be applied to any ball through subsequent iterations. The intuition behind this bound is the well-known relationship between increasing the density of sampling and the increased local flatness of the surface approximation. Specifically, shrinkage decreases as the density increases. As it turns out, violations of the deep coverage condition C4 are the main cause for refinement after shrinking to eliminate slivers. The termination of the algorithm can be guaranteed by accepting a set of balls with

no half-covered seeds as long as all boundary points are only α'-deeply covered, for some $\alpha' < \alpha$. As we prove in the supplemental materials, the smoothness of the input surface per the parameter $\theta^\flat$ guarantees that shrinkage eventually falls below a threshold that cannot violate $\frac{\alpha}{2}$-deep coverage.

Practical Variant.—Our implementation always reruns RMPS to recover α-deep coverage. In what follows, we argue that this implementation terminates with high probability by combining the bounds on shrinkage with the stability of deep coverage as a distribution. In our experiments, VoroCrust always terminates with all slivers eliminated successfully while avoiding excessive refinement; see Section 3. In the unlikely event that sliver elimination fails to terminate in a constant number of iterations, set to 100, our implementation restarts in a *safe mode* which accepts $\frac{\alpha}{2}$-deep coverage to guarantee termination; we never encountered such cases.

Shrinkage Ratio.—



Fix a triplet t and let $g^\uparrow$ and $g^\downarrow$ denote the intersection points of its bounding spheres, such that t has a half-covered seed due to a fourth ball b_q. Assume w.l.o.g. that $g^\downarrow \in b_q$ while $g^\uparrow \notin b_q$, i.e., $\|q - g^\downarrow\| < r_q$ while $\|q - g^\uparrow\| \geq r_q$; see the inset. To resolve the half-covered seed, the algorithm shrinks b_q by setting its radius to $\|q - g^\downarrow\|$. Hence, the shrinkage is $r_q - \|q - g^\downarrow\| > 0$. As violations of α-deep coverage after shrinking are the main cause for further refinement, we consider shrinkage as a ratio of the orignal radius which we denote by Δ. The above inequalities imply the following bound:

$\Delta = \frac{r_q - \|q - g^\downarrow\|}{r_q} \leq \frac{\|q - g^\uparrow\| - \|q - g^\downarrow\|}{\|q - g^\downarrow\|} = \frac{\|q - g^\uparrow\|}{\|q - g^\downarrow\|} - 1$. In particular, as $\frac{\|q - g^\uparrow\|}{\|q - g^\downarrow\|}$ approaches 1, α-deep coverage is less likely to be violated after shrinking. Specifically, if $\Delta \leq \frac{\alpha}{\alpha - 2}$, then $\frac{\alpha}{2}$-deep coverage holds. Assuming the input $\mathcal{T}$ is sufficiently smooth per $\theta^\flat$, this observation

guarantees the termination of the algorithm if $\frac{\alpha}{2}$-deep coverage is accepted; see the supplemental materials for the proof and further discussion.

Decaying Shrinkage and Violations.—Subsequent invocations of RMPS in the course of sliver elimination increase the density of sampling. A consequence of the ball conditions maintained by RMPS is that the radii of overlapping balls get smaller. In particular, the deviation in normals at the centers of overlapping balls gets smaller, which is equivalent to enforcing the smooth overlap condition C2 with a smaller angle threshold. Intuitively, the neighborhood of each sample becomes *nearly flat*. This flatness increases the ratio $\frac{\|q-g^{\uparrow}\|}{\|q-g^{\downarrow}\|}$ for all nearby samples q, which reduces the shrinkage ratio Δ and restricts the potential locations of new samples that create new slivers. It follows that the percentage of triplets with half-covered seed pairs decays rapidly; see Figure 9(right).

Deep-coverage Distribution.—Let f_i be a function that maps each $x \in \mathcal{T}$ to $\max\left\{1-\frac{\|x-p\|}{r_p} \mid b_p \in \mathcal{B}_{i,x}\right\}$ where $\mathcal{B}_{i,x}$ is the subset of balls containing x at iteration i. We use the family of functions $\{f_i\}$ to define the deep-coverage distribution as $F_i(\alpha) = \Pr[f_i(x) \leq \alpha \mid x \in \mathcal{T}]$ with $\alpha \in [0, 1]$. We estimate F_i by the empirical distribution function over 100 bins using independent random samples of 10^6 points. Figure 9(left) shows the evolution of the deep-coverage distribution through the first invocation of RMPS until convergence. Every subsequent invocation of RMPS, following shrinking for sliver elimination, converges to a nearly identical distribution. Related aspects of the distributions of MPS samplings were analyzed [Mitchell et al. 2012], which are consistent with our experiments[1]; see the supplemental materials for further examples and discussion. As seen in Figure 9(middle), shrinking for sliver elimination initially violates α-deep coverage, per C4 requiring a fixed $\alpha \approx 0.13$, but causes no such violations over the last few iterations. The combination of decaying shrinkage and the stability of deep coverage as a distribution bounds the probability of such violations. It follows that subsequent invocations of RMPS are less likely to introduce new balls to recover α-deep coverage. As a result, the number of newly created slivers per iteration decays rapidly; see Figure 9(right). Hence, the total number of slivers encountered by the algorithm is bounded in expectation, which implies termination in a finite number of steps with high probability.

2.5 Volume Meshing

Once the refinement process terminates, the set of balls $\mathcal{B}$ is fixed and a conforming surface mesh can be generated. To further decompose the interior into a set of graded Voronoi cells, additional weighted samples $\mathcal{S}^{\downarrow\downarrow}$ are generated in the interior of the domain. Similar to $\mathcal{B}$, the balls corresponding to interior samples are required to satisfy the L-Lipschitzness condition. Standard MPS may be used for sampling the interior. However, to reduce the memory footprint of this step, the spoke-darts algorithm [Mitchell et al. 2018] is used instead following a lightweight initialization phase using standard dart-throwing; see the

[1]The total variation distance [DasGupta 2008] between the empirical distributions obtained through all subsequent iterations is at most 0.02.

appendix for more details. Alternatively, the interior samples may be chosen as the vertices of a structured lattice. This can be used to output a hex-dominant mesh conforming to the surface; see Figure 5(f). The quality of the volume mesh can be further improved by applying CVT optimization to the set of interior seeds; see Figure 5(d).

2.6 Meshing 2D Domains

The proposed VoroCrust algorithm can readily be applied to the decomposition of 2D domains into conforming Voronoi meshes. As illustrated in Figure 4, the seed placement strategy can be applied in 2D given a suitable union of balls. The refinement strategy described in this section can easily be applied to generate such a union of balls by regarding the 2D boundary as a set of creases embedded in 3D. In particular, assuming the 2D boundary is available as a set of line segments or a planar straight-line graph (PSLG) as common in 2D meshing, the input segments can be mapped to 3D by adding a third coordinate, e.g., $z = 0$, to all end points. The ball conditions and refinement process for the protection of sharp features, as defined in Section 2.3, guarantee a union of balls that approximates the embedded 2D boundary.

Such a union of balls can be used to place Voronoi seeds in 2D as follows. First, all balls are projected onto the 2D plane as circles centered along the boundary. Then, the pairs of intersection points between consecutive circles are computed. Recalling that the edge balls protecting any given crease may only overlap consecutive balls along the same crease, these pairs of intersection points are well-defined. Once the intersection pairs are obtained, the algorithm places Voronoi seeds across the 2D boundary and proceeds to sample additional seeds in the interior of the 2D domain. Figure 10 shows a number of conforming 2D meshes obtained by a 2D implementation of the VoroCrust algorithm.

3 RESULTS

We demonstrate the capabilities of the VoroCrust algorithm and study the impact of input parameters. Then, we compare against the work of Yan et al. [2010] as a representative of state-of-the-art clipping-based methods. All experiments were conducted on a Mac Pro machine with a 3.5 GHz 6-Core Intel Xeon E5 processor and 32 GB of RAM.

Robustness and Quality.

We test VoroCrust on a variety of models exhibiting different challenges ranging from smooth models with detailed features and narrow regions as in Figure 11, to sharp features with curvature and holes, and even non-manifold boundaries as in Figure 12.

The quality of the surface mesh is measured by the minimum triangle quality[2] Q_{min}, as well as the percentage of triangles with angles less than 30° or greater than 90° and vertices with valence 6. We report approximation errors in terms of the Hausdorff error d_H (normalized by the diameter of the bounding box) and the root mean squared distance d_{RMS}. The quality of the volume mesh is measured by the maximum aspect ratio[3] ρ_{max}, which is often realized by

[2]Triangle quality is defined as $\frac{6S}{\sqrt{3}hP}$, where S is the area, h is the longest edge length, and P is half the perimeter.

cells incident to the surface. The running times taken to generate $\mathcal{S}^{\updownarrow}$ and $\mathcal{S}^{\downarrow\downarrow}$, denoted $T^{\updownarrow}$ and $T^{\downarrow\downarrow}$, are reported in seconds. Meshes were generated from VoroCrust seeds using Voro^{++} [Rycroft 2009].

We encountered no issues with any of the models, which demonstrates the robustness of the algorithm and its implementation. We set $\theta^\sharp$ to 60° for smooth models, and choose an appropriate value of $\theta^\sharp$ for models with sharp features. The value of L was fixed at 0.25 for all inputs. We note that the output surface meshes are of high quality per the minimum triangle quality and angle bounds, while achieving small approximation errors. The demonstrated quality of VoroCrust output, with no skinny elements, is in agreement with the theoretical guarantees established for an abstract version of the algorithm [Abdelkader et al. 2018]. Additional results on a variety of models are provided in the supplemental materials.

Parameters.

We start by studying the impact of L on the complexity of the output surface mesh and the running time of the algorithm. Figure 13 demonstrates this impact on the Joint model. The results of this experiment demonstrate the influence of L on the level of refinement per the number of balls in $\mathcal{B}$ generated by the algorithm. In particular, smaller values of L lead to higher refinement. On the other hand, larger values of L slow down the algorithm due to the increased size of ball neighborhoods resulting in processing a larger number of balls for various tasks; see Section A.3. This behavior of the algorithm in terms of L is consistent for different values of $\theta^\sharp$ as can be seen in Figure 13.

Next, we study the impact of varying both L and $\theta^\sharp$. We chose a relatively simple smooth model to better assess the degradation in surface approximation. Figure 16 illustrates VoroCrust output on the Goat model for 5×5 combinations of parameter settings. As shown earlier, smaller values of L result in more regular meshes with superior element quality per the minimum triangle angle. On the other hand, the parameter $\theta^\sharp$ controls the surface approximation. Namely, higher values of $\theta^\sharp$ result in higher Hausdorff errors.

Finally, we study the impact of the input sizing field sz on the multi-layered nested spheres models. Figure 14 shows how sz can be used to directly control ball radii to enforce further refinement. The default setting of $sz = \infty$ incurs the minimum level of refinement required by the geometry of the domain according to the quality requirements indicated by the parameters L and $\theta^\sharp$. We note that sz can be specified as a spatially varying sizing field.

In summary, this study demonstrates the flexibility of the VoroCrust algorithm to accommodate a wide range of parameter settings that cater to the requirements of different applications. In particular, the set of parameters provided allows the user to trade-off the quality of the surface mesh, approximation error, output complexity, and running time.

[3]Aspect ratio is defined as the ratio between the radius of the smallest circumscribing sphere to the radius of the largest inscribed sphere.

Comparison.

We compare against the restricted Voronoi diagram (RVD) of Yan et al. [2010] as a representative of state-of-the-art polyhedral meshing algorithm based on clipped Voronoi cells. As shown in Figure 1, VoroCrust achieves superior quality in terms of the surface mesh, where the RVD produces an imprint of the input mesh with many small facets. In particular, by examining the ratio of the shortest to longest edge length per surface facet, it is clear that RVD results in many skinny facets which can be problematic for many applications. Moreover, RVD results in non-convex cells for the non-convex model in Figure 2. In contrast, VoroCrust output conforms to the boundary with true Voronoi cells, which are guaranteed to be convex, while achieving much better quality of surface elements. Additional comparisons against RVD are provided in the supplemental materials.

4 LIMITATIONS

The main limitation of the presented algorithm is the possible presence of short Voronoi edges in the interior of the output mesh, which can lead to small time steps in numerical simulations significantly increasing their cost. To eliminate such short edges, mesh improvement techniques may be applied as postprocessing [Abdelkader et al. 2017b; Sieger et al. 2010].

Another limitation is the requirement that the input triangulation is a faithful approximation of the domain. This inhibits the application of this approach to implicit forms [Wang et al. 2016], noisy inputs [Mederos et al. 2005], or unclean geometries [Attene et al. 2013]. In particular, the algorithm does not fill holes or undesirable cracks in non-watertight inputs [Kumar et al. 2008]. Nonetheless, VoroCrust readily handles surfaces with boundary as shown in Figure 15.

Finally, the isotropic nature of the proposed sampling process may result in an unnecessarily large number of cells in narrow regions. For such geometries, boundary layers of elongated cells enable higher fidelity near the boundary [Garimella and Shephard 2000; Loseille and Löhner 2013]. In cases of strong anisotropy, aligning the cells, e.g., to the eigenvectors of a Hessian [Budninskiy et al. 2016; Fu et al. 2014], better captures the variation of physical quantities.

5 CONCLUSION

Voronoi cells provide a competitive alternative to traditional mesh elements with many desired features that follow naturally from their definition, e.g., convexity, convex planar facets, and orthogonal duals provided by the corresponding Delaunay elements. As general polyhedral elements, Voronoi cells enjoy greater degrees of freedom that enable better mesh connectivity and the ability to conform to complex geometries undergoing large deformations. What hindered their wide-scale adoption is the lack of a robust Voronoi meshing algorithm that can handle broad classes of domains exhibiting arbitrary curved boundaries and sharp features.

We developed the VoroCrust algorithm to fill this gap. VoroCrust is based on a well-principled mirroring approach combined with state-of-the-art techniques for automatic sizing estimation to mesh piecewise-smoothed complexes. We proved strong theoretical guarantees on the correctness of the VoroCrust algorithm and the quality of its output, as demonstrated through a variety of models. By conducting an extensive comparison against state-of-the-art polyhedral meshing methods based on clipped Voronoi cells, we established the advantage of VoroCrust output.

For future work, we consider the generation of boundary layers and taking anisotropy into account. We believe that VoroCrust refinement can be extended to accommodate additional requirements catering to the quality of the cells while preserving the surface approximation. To match the quality of the output surface mesh, improving the quality of the volume mesh by eliminating short Voronoi edges possibly present in the interior is particularly important. Such short edges arise when the distance from a Voronoi vertex v to its $d+1$ closest Voronoi seeds is only slightly less than its distance to the $(d+2)^{th}$ nearest Voronoi seed s. In other words, the seed s lies close to the *Delaunay sphere* centered at v. A potential approach to avoid such configurations is to define a *buffer zone* to penalize the placement of Voronoi seeds that result in short Voronoi edges. The buffer zone can be defined as a thickening of the Delaunay sphere into a spherical shell whose thickness is a small fraction of the radius. We are currently exploring a restricted sampling technique based on this idea.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research (ASCR), Applied Mathematics Program. Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

REFERENCES

Abdelkader Ahmed, Bajaj Chandrajit L., Ebeida Mohamed S., Mahmoud Ahmed H., Mitchell Scott A., Owens John D., and Rushdi Ahmad A. 2018 Sampling conditions for conforming Voronoi meshing by the VoroCrust algorithm. In 34th International Symposium on Computational Geometry (SoCG 2018) (Leibniz International Proceedings in Informatics (LIPIcs)), Vol. 99 1:1–1:16.

Abdelkader Ahmed, Bajaj Chandrajit L., Ebeida Mohamed S., and Mitchell Scott A. 2017a A seed placement strategy for conforming Voronoi meshing. In 29th Canadian Conference on Computational Geometry (CCCG) 95–100.

Abdelkader Ahmed, Mahmoud Ahmed H., Rushdi Ahmad A., Mitchell Scott A., Owens John D., and Ebeida Mohamed S. 2017b A constrained resampling strategy for mesh improvement. Computer Graphics Forum 36, 5 (2017), 189–201.

Akbarzadeh Masoud, Van Mele Tom, and Block Philippe. 2015 On the equilibrium of funicular polyhedral frames and convex polyhedral force diagrams. Computer-Aided Design 63 (2015), 118–128.

Alliez Pierre, Cohen-Steiner David, Yvinec Mariette, and Desbrun Mathieu. 2005 Variational tetrahedral meshing. ACM Trans. Graph 24, 3 (2005), 617–625.

Amenta Nina and Bern Marshall. 1999 Surface reconstruction by Voronoi filtering. Discrete & Computational Geometry 22, 4 (1999), 481–504.

Amenta Nina, Choi Sunghee, and Kolluri Ravi Krishna. 2001 The power crust, unions of balls, and the medial axis transform. Computational Geometry 19, 2 (2001), 127–153.

Amenta N. and Kolluri R-K 2001 The medial axis of a union of balls. Computational Geometry 20, 1 (2001), 25–37. Selected papers from the 12th Annual Canadian Conference.

Amresh Ashish, Farin Gerald, and Razdan Anshuman. 2003 Adaptive subdivision schemes for triangular meshes. In Hierarchical and Geometrical Methods in Scientific Visualization. 319–327.

Attene Marco, Campen Marcel, and Kobbelt Leif. 2013 Polygon mesh repairing: an application perspective. ACM Comput. Surv 45, 2, Article 15 (2013), 15:1–15:33 pages.

Aurenhammer Franz, Klein Rolf, and Lee Der-Tsai. 2013 Voronoi Diagrams and Delaunay Triangulations. Vol. 8 World Scientific.

da Veiga L. Beir0ão, Brezzi F, Cangiani A, Manzini G, Marini LD, and Russo A. 2013 Basic principles of virtual element methods. Mathematical Models and Methods in Applied Sciences 23, 01 (2013), 199–214.

Bishop JE 2014 A displacement-based finite element formulation for general polyhedra using harmonic shape functions. Internat. J. Numer. Methods Engrg 97, 1 (2014), 1–31.

Bochev Pavel B. and Hyman James M. 2006 Principles of mimetic discretizations of differential operators. In Compatible Spatial Discretizations. 89–119.

Bonduà S, Bortolotti V, Macini P, Mesini E, and Vasini EM 2017 3D Voronoi pre- and post-processing tools for using the TOUGH2 family of numerical simulator for hydrocarbon gas migration. In Offshore Mediterranean Conference.

Budninskiy Max, Liu Beibei, Fernando de Goes, Tong Yiying, Alliez Pierre, and Desbrun Mathieu. 2016 Optimal Voronoi tessellations with Hessian-based anisotropy. ACM Trans. Graph 35, 6 (2016), 242:1–242:12.

Chen Zhili, Yao Miaojun, Feng Renguo, and Wang Huamin. 2014 Physics-inspired adaptive fracture refinement. ACM Trans. Graph 33, 4 (2014), 113:1–113:7.

Cheng Siu-Wing, Dey Tamal K., and Ramos Edgar A. 2010 Delaunay refinement for piecewise smooth complexes. Discrete & Computational Geometry 43, 1 (2010), 121–166.

Cheng Siu-Wing, Dey Tamal K, and Shewchuk Jonathan. 2012 Delaunay Mesh Generation. CRC Press.

Cheng Siu-Wing and Poon Sheung-Hung. 2006 Three-dimensional Delaunay mesh generation. Discrete & Computational Geometry 36, 3 (2006), 419–456.

Clausen Pascal, Wicke Martin, Shewchuk Jonathan R., and O'Brien James F. 2013 Simulating liquids and solid-liquid interactions with Lagrangian meshes. ACM Trans. Graph 32, 2 (2013), 17:1–17:15.

Cohen-Steiner D, de Verdière E-C, and Yvinec M. 2002 Conforming Delaunay triangulations in 3D. In Proceedings of the Eighteenth Annual Symposium on Computational Geometry (SCG '02) 199–208.

DasGupta Anirban. 2008 Metrics, information theory, convergence, and Poisson approximations In Asymptotic Theory of Statistics and Probability. Springer, Chapter 2, 19–34.

Dawes AS 2017 Three-dimensional multi-material polyhedral method for diffusion. Computers & Fluids 156 (2017), 485–495.

Desbrun Mathieu, Kanso Eva, and Tong Yiying. 2008 Discrete differential forms for computational modeling In Discrete Differential Geometry. Oberwolfach Seminars, Vol. 38 Birkhauser Basel, 287–324.

Dey Tamal K. and Levine Joshua A. 2009 Delaunay meshing of piecewise smooth complexes without expensive predicates. Algorithms 2, 4 (2009), 1327–1349.

Dey Tamal K. and Ray Tathagata. 2010 Polygonal surface remeshing with Delaunay refinement. Engineering with Computers 26, 3 (2010), 289–301.

Duan Liuyun and Lafarge Florent. 2015 Image partitioning into convex polygons. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 3119–3127.

Ebeida Mohamed S. and Mitchell Scott A. 2012 Uniform random Voronoi meshes. In Proceedings of the 20th International Meshing Roundtable 273–290.

Ebeida Mohamed S., Mitchell Scott A., Patney Anjul, Davidson Andrew A., and Owens John D. 2012 A simple algorithm for maximal Poisson-disk sampling in high dimensions. Computer Graphics Forum 31, 2 (2012), 785–794.

Elcott Sharif, Tong Yiying, Kanso Eva, Schröder Peter, and Desbrun Mathieu. 2007 Stable, circulation-preserving, simplicial fluids. ACM Trans. Graph 26, 1 (2007).

Engwirda Darren. 2018 Generalised primal-dual grids for unstructured co-volume schemes. J. Comput. Phys 375 (2018), 155–176.

Freeman CM, Boyle KL, Reagan M, Johnson J, Rycroft C, and Moridis GJ 2014 MeshVoro: A three-dimensional Voronoi mesh building tool for the TOUGH family of codes. Computers & Geosciences 70 (2014), 26–34.

Fu Xiao-Ming, Liu Yang, Snyder John, and Guo Baining. 2014 Anisotropic simplicial meshing using local convex functions. ACM Trans. Graph 33, 6, Article 182 (2014), 11 pages.

Gain Arun L., Talischi Cameron, and Paulino Glaucio H. 2014a On the virtual element method for three-dimensional linear elasticity problems on arbitrary polyhedral meshes. Computer Methods in Applied Mechanics and Engineering 282 (2014), 132–160.

Gain Arun L., Talischi Cameron, and Paulino Glaucio H. 2014b On the virtual element method for three-dimensional linear elasticity problems on arbitrary polyhedral meshes. Computer Methods in Applied Mechanics and Engineering 282 (2014), 132–160.

Garimella Rao V., Kim Jibum, and Berndt Markus. 2014 Polyhedral mesh generation and optimization for non-manifold domains. In Proceedings of the 22nd International Meshing Roundtable 313–330.

Garimella RV and Lipnikov K. 2011 Solution of the diffusion equation in multi-material domains by sub-division of elements along reconstructed interfaces. International Journal for Numerical Methods in Fluids 65, 11–12 (2011), 1423–1437.

Garimella Rao V. and Shephard Mark S. 2000 Boundary layer mesh generation for viscous flow simulations. Internat. J. Numer. Methods Engrg 49, 1–2 (2000), 193–218.

de Goes Fernando, Memari Pooran, Mullen Patrick, and Desbrun Mathieu. 2014 Weighted triangulations for geometry processing. ACM Trans. Graph 33, 3 (2014), 28:1–28:13.

Guo Jianwei, Yan Dong-Ming, Jia Xiaohong, and Zhang Xiaopeng. 2015 Efficient maximal Poisson-disk sampling and remeshing on surfaces. Computers & Graphics 46 (2015), 72–79.

Harlow Francis H. and Welch J. Eddie. 1965 Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. The Physics of Fluids 8, 12 (1965), 2182–2189.

Hu Litang, Zhang Keni, Cao Xiaoyuan, Li Yi, and Guo Chaobin. 2016 IGMESH: A convenient irregular-grid-based pre- and post-processing tool for TOUGH2 simulator. Computers & Geosciences 95 (2016), 11–17.

Hu Yixin, Zhou Qingnan, Gao Xifeng, Jacobson Alec, Zorin Denis, and Panozzo Daniele. 2018 Tetrahedral meshing in the wild. ACM Trans. Graph 37, 4, Article 60 (2018), 60:1–60:14 pages.

Kikinzon Evgeny, Kuznetsov Yuri, Lipnikov Konstatin, and Shashkov Mikhail. 2017 Approximate static condensation algorithm for solving multi-material diffusion problems on meshes non-aligned with material interfaces. J. Comput. Phys 347 (2017), 416–436.

Kim Seong-Kyun, Bae Gwang-Ok, and Lee Kang-Kun. 2015 Improving accuracy and flexibility of numerical simulation of geothermal heat pump systems using Voronoi grid refinement approach. Geosciences Journal 19, 3 (2015), 527–535.

Kumar Amitesh, Shih Alan, Ito Yasushi, Ross Douglas, and Soni Bharat. 2008 A Hole-filling Algorithm Using Non-uniform Rational B-splines. In Proceedings of the 16th International Meshing Roundtable 169–182.

Lichtner Peter C., Hammond Glenn E., Lu Chuan, Karra Satish, Bisht Gautam, Andre Benjamin, Mills Richard, and Kumar Jitendra. 2015 PFLOTRAN user manual: A massively parallel reactive flow and transport model for describing surface and subsurface processes Technical report LA-UR-15–20403 (2015). Los Alamos National Lab; Sandia National Lab; Lawrence Berkeley National Lab; Oak Ridge National Lab; OFM Research.

Lipnikov Konstantin, Manzini Gianmarco, and Shashkov Mikhail. 2014 Mimetic finite difference method. J. Comput. Phys 257 (2014), 1163–1227. Physics-compatible numerical methods.

Loseille Adrien and Lohner Rainald. 2013 Robust boundary layer mesh generation. In Proceedings of the 21st International Meshing Roundtable 493–511.

Macneal RH 1953 An asymmetrical finite difference network. Quart. Appl. Math 11, 3 (1953), 295–310.

Manzini Gianmarco, Russo Alessandro, and Sukumar N. 2014 New perspectives on polygonal and polyhedral finite element methods. Mathematical Models and Methods in Applied Sciences 24, 08 (2014), 1665–1699.

Martin Sebastian, Kaufmann Peter, Botsch Mario, Wicke Martin, and Gross Markus. 2008 Polyhedral finite elements using harmonic basis functions. Computer Graphics Forum 27, 5 (2008), 1521–1529.

Maxwell James Clerk. 1870 On reciprocal figures, frames, and diagrams of forces. Transactions of the Royal Society of Edinburgh 26, 1 (1870), 1–40.

Mederos Boris, Amenta Nina, Velho Luiz, and de Figueiredo Luiz Henrique. 2005 Surface reconstruction from noisy point clouds. In Proceedings of the Third Eurographics Symposium on Geometry Processing (SGP '05).

Mercat Christian. 2001 Discrete Riemann surfaces and the Ising model. Communications in Mathematical Physics 218, 1 (2001), 177–216.

Miller G, Talmor D, and Teng S-H 1999 Data generation for geometric algorithms on non-uniform distributions. International Journal of Computational Geometry and Applications 09, 06 (1999), 577–597.

Mitchell Scott A., Ebeida Mohamed S., Awad Muhammad A., Park Chonhyon, Patney Anjul, Rushdi Ahmad A., Swiler Laura P., Manocha Dinesh, and Wei Li-Yi. 2018 Spoke-darts for high-dimensional blue-noise sampling. ACM Trans. Graph 37, 2 (2018), 22:1–22:20.

Mitchell Scott A, Rand Alexander, Ebeida Mohamed S, and Bajaj Chandrajit. 2012 Variable radii Poisson-disk sampling. In Proceedings of the 24th Canadian Conference on Computational Geometry (CCCG).

Mullen Patrick, Memari Pooran, de Goes Fernando, and Desbrun Mathieu. 2011 HOT: hodge-optimized triangulations. ACM Trans. Graph 30, 4 (2011), 103:1–103:12.

Murphy M, Mount D, and Gable C. 2001 A point-placement strategy for conforming Delaunay tetrahedralization. International Journal of Computational Geometry & Applications 11, 06 (2001), 669–682.

Nicolaides Roy A. and Wu Xiaonan. 1997 Covolume solutions of three-dimensional div-curl equations. SIAM J. Numer. Anal 34, 6 (1997), 2195–2203.

Novak Igor L., Gao Fei, Choi Yung-Sze, Resasco Diana, Schaff James C., and Slepchenko Boris M. 2007 Diffusion on a curved surface coupled to diffusion in the volume: Application to cell biology. J. Comput. Phys 226, 2 (2007), 1271–1290. [PubMed: 18836520]

Oudot Steve, Rineau Laurent, and Yvinec Mariette. 2010 Meshing volumes with curved boundaries. Engineering with Computers 26, 3 (2010), 265–279.

Peric M. and Ferguson S. Spring 2005 The advantage of polyhedral meshes. Dynamics - Issue 24 (Spring 2005), 4–5. The customer magazine of the CD-adapco Group, currently maintained by Siemens at http://siemens.com/mdx The issue is available at http://mdx2.plm.automation.siemens.com/magazine/dynamics-24 (accessed January 17, 2019).

Perot Blair. 2000 Conservation properties of unstructured staggered mesh schemes. J. Comput. Phys 159, 1 (2000), 58–89.

Pruess K. 1991 TOUGH2: A general-purpose numerical simulator for multiphase fluid and heat flow. NASA STI/Recon Technical Report N 92 (1991).

Pruess K. 2004 The TOUGH codes — a family of simulation tools for multiphase flow and transport processes in permeable media. Vadose Zone Journal 3 (2004).

Rand Alexander and Walkington Noel. 2008 3D Delaunay refinement of sharp domains without a local feature size oracle. In Proceedings of the 17th International Meshing Roundtable 37–54.

Rand Alexander and Walkington Noel. 2009 Collars and intestines: practical conforming Delaunay refinement. In Proceedings of the 18th International Meshing Roundtable.

Rankine William John Macquorn. 1864 Principle of the equilibrium of polyhedral frames. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 27, 180 (1864), 92–.

Rycroft Chris. 2009 Voro++: A Three-Dimensional Voronoi Cell Library in C++. Chaos 19, 4 (2009), 041111 Software available online at http://math.lbl.gov/voro++/. [PubMed: 20059195]

Klemetsdal ØS, Berge RL, Lie Knut-Andreas, Nilsen HM, and Møyner Olav. 2017 Unstructured gridding and consistent discretizations for reservoirs with faults and complex wells. In Society of Petroleum Engineers Reservoir Simulation Conference.

Shewchuk Jonathan. 2002 What is a good linear finite element? interpolation, conditioning, anisotropy, and quality measures. preprint. University of California at Berkeley.

Si H, Gartner K, and Fuhrmann J. 2010 Boundary conforming Delaunay mesh generation. Computational Mathematics and Mathematical Physics 50, 1 (2010), 38–53.

Sieger Daniel, Alliez Pierre, and Botsch Mario. 2010 Optimizing Voronoi diagrams for polygonal finite element computations. Springer, 335–350.

Sukumar N and Bolander JE. 2009 Voronoi-based interpolants for fracture modelling. In Tessellations in the Sciences: Virtues, Techniques and Applications of Geometric Tilings.

Talischi Cameron, Paulino Glaucio H., Pereira Anderson, and Menezes Ivan F. M. 2013 Polygonal finite elements for topology optimization: A unifying paradigm. Internat. J. Numer. Methods Engrg 82, 6 (2013), 671–698.

Tournois Jane, Wormser Camille, Alliez Pierre, and Desbrun Mathieu. 2009 Interleaving Delaunay refinement and optimization for practical isotropic tetrahedron mesh generation. ACM Trans. Graph 28, 3 (2009), 75:1–75:9.

VanderZee Evan, Hirani Anil N., Guoy Damrong, and Ramos Edgar A. 2010 Well-Centered triangulation. SIAM Journal on Scientific Computing 31, 6 (2010), 4497–4523.

Wang Li, Hétroy-Wheeler Franck, and Boyer Edmond. 2016 On volumetric shape reconstruction from implicit forms. In Computer Vision – ECCV 2016 173–188.

Warren Joe, Schaefer Scott, Hirani Anil N., and Desbrun Mathieu. 2007 Barycentric coordinates for convex sets. Advances in Computational Mathematics 27, 3 (2007), 319–338.

Wicke Martin, Botsch Mario, and Gross Markus. 2007 A finite element method on convex polyhedra. Computer Graphics Forum 26, 3 (2007), 355–364.

Wicke Martin, Ritchie Daniel, Klingner Bryan M., Burke Sebastian, Shewchuk Jonathan R., and O'Brien James F. 2010 Dynamic local remeshing for elastoplastic simulation. ACM Trans. Graph 29, 4 (2010), 49:1–49:11.

Wojtan Chris, Thurey Nils, Gross Markus, and Turk Greg. 2009 Deforming meshes that split and merge. ACM Trans. Graph 28, 3 (2009), 76:1–76:10.

Wu Jun, Westermann Rudiger, and Dick Christian. 2015 A survey of physically based simulation of cuts in deformable bodies. Computer Graphics Forum 34, 6 (2015), 161–187.

Yan Dong-Ming, Wang Wenping, Lévy Bruno, and Liu Yang. 2010 Efficient computation of 3D clipped Voronoi diagram In Advances in Geometric Modeling and Processing. Lecture Notes in Computer Science, Vol. 6130 Springer, 269–282.

Yan Dong-Ming and Wonka Peter. 2013 Gap processing for adaptive maximal Poisson-disk sampling. ACM Transactions on Graphics 32, 5 (2013), 148:1–148:15.

CCS Concepts:

- **Computing methodologies → Computer Graphics**.

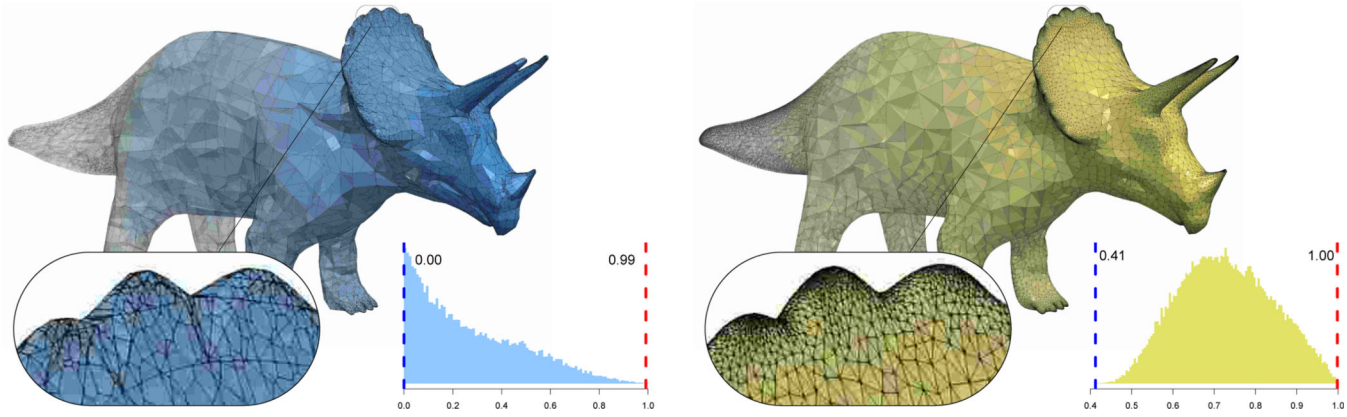


Fig. 1.

State-of-the-art methods for conforming Voronoi meshing clip Voronoi cells at the bounding surface. The Restricted Voronoi Diagram [Yan et al. 2010] (left) is sensitive to the input tessellation and produces surface elements of very low quality, per the shortest-to-longest edge ratio distribution shown in the inset. In contrast, VoroCrust (right) generates an unclipped Voronoi mesh conforming to a high-quality surface mesh.

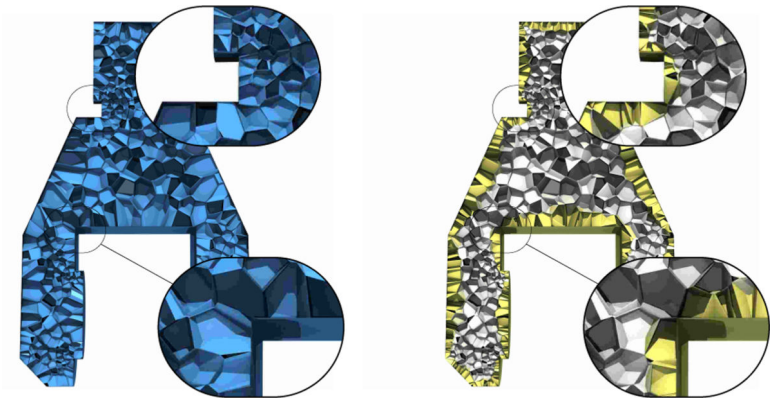


Fig. 2.

State-of-the-art clipping [Yan et al. 2010] may create non-convex cells (left). In contrast, VoroCrust always produces true Voronoi cells conforming to the boundary (right).

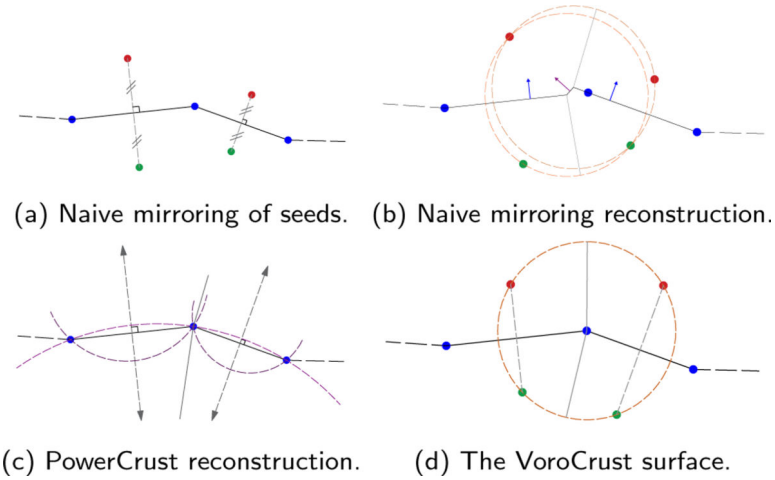


Fig. 3.

Voronoi-based surface reconstruction. Naive mirroring (a) results in bad normals (b) due to Voronoi facets between non-mirrored seeds. PowerCrust avoids this issue by placing weighted seeds on the medial axis away from the surface (c). VoroCrust reduces the misaligned facet to a vertex (d) using unweighted seeds.

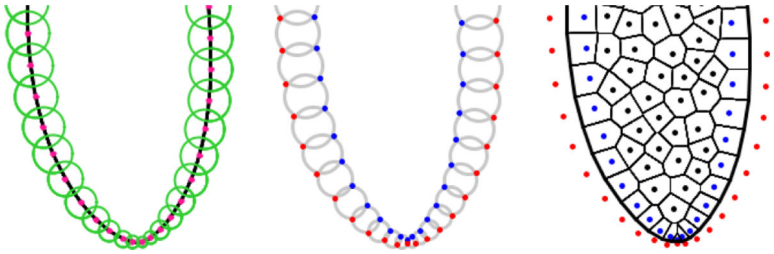


Fig. 4.

VoroCrust summary: (left) Cover the boundary by a union of balls, (middle) place pairs of Voronoi seeds where balls intersect to capture and isolate the boundary, and finally (right) seed the interior.

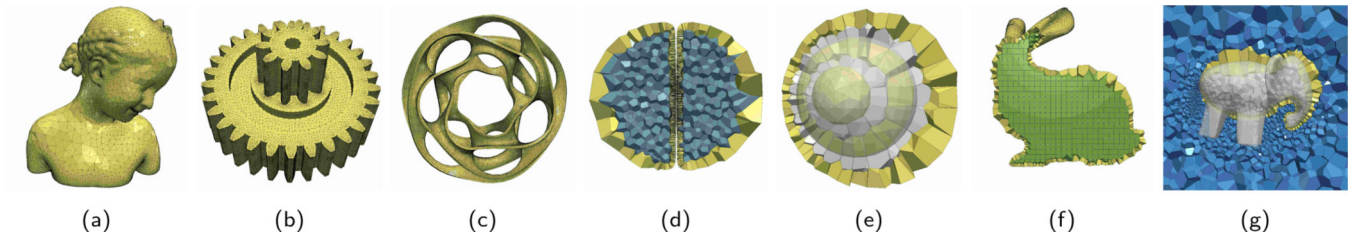


Fig. 5.

VoroCrust can handle inputs containing both smooth (a), sharp features (b), and complex topology (c), with possibly multiple components (d), and multi-layers interfacing different types of material (e). To decompose the volume, VoroCrust has the flexibility of using seeds on a lattice (f), or generated randomly (g), or optimized by CVT (d).

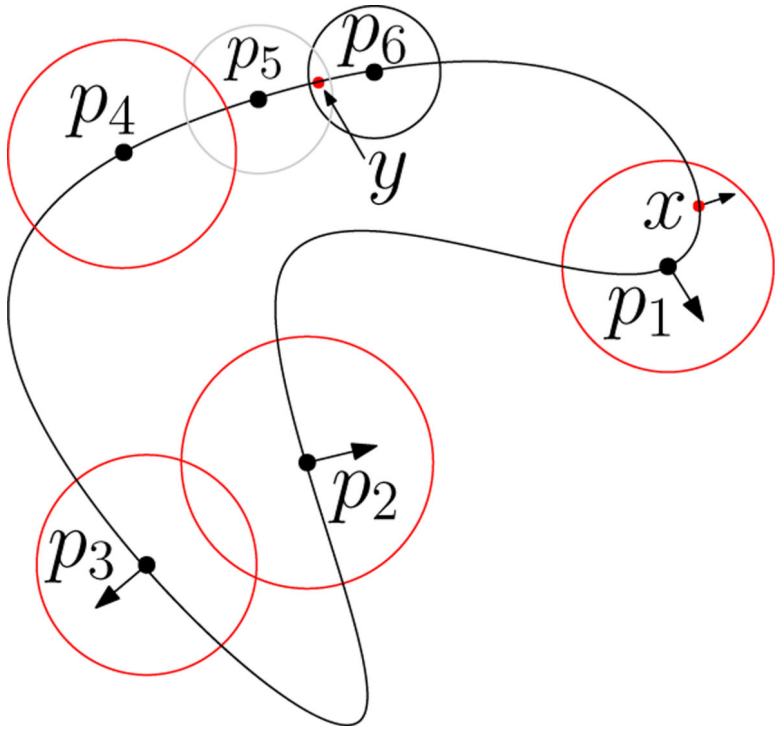


Fig. 6.

Ball conditions. C1 is violated for x by b_{p1}. C2 is violated for b_{p2} and b_{p3}. C3 is violated by b_{p4}. C4 is violated for y.

Fig. 7.

The three phases of VoroCrust refinement demonstrated on the Fandisk model: protection by corner balls (left) followed by edge balls (center), and finally coverage by surface balls (right).

Fig. 8.

Sliver elimination: (left) A quartet of balls centered at four samples (black) with four half-covered seeds yielding a Steiner vertex (pink) with four incident facets. (right) Shrinking resolves half-covered seeds and eliminates the Steiner vertex yielding only two facets. Refer to the supplemental materials for more details about this example.

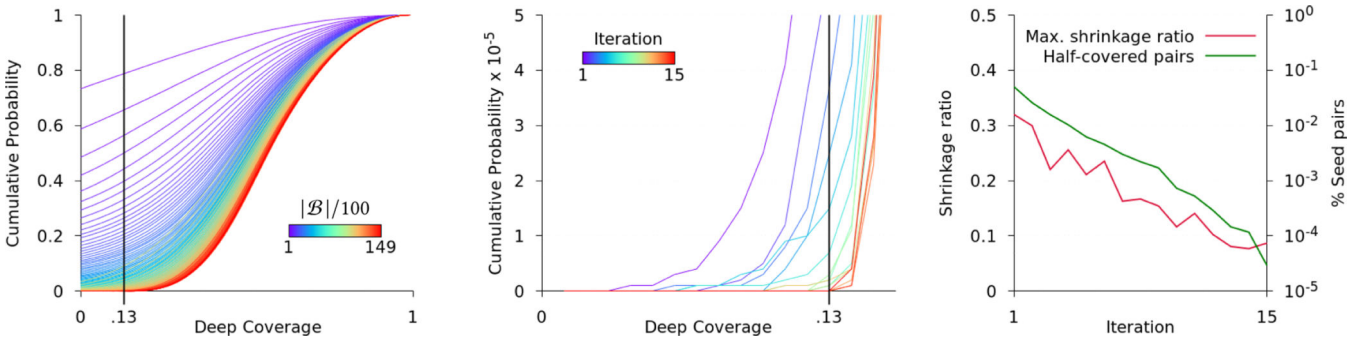


Fig. 9.

Empirical analysis of sliver elimination using the Bimba model: (left) evolution of the deep-coverage distribution through the first invocation of RMPS as $\mathscr{B}$ grows in increments of 100 balls, (middle) sliver elimination executes 15 iterations where shrinking eventually ceases to violate α-deep coverage, (right) the refinement incurred by sliver elimination decreases the maximum shrinkage ratio applied in subsequent iterations. As a result, the number of newly created slivers, measured by the percentage of triplets with half-covered seed pairs, decays rapidly.

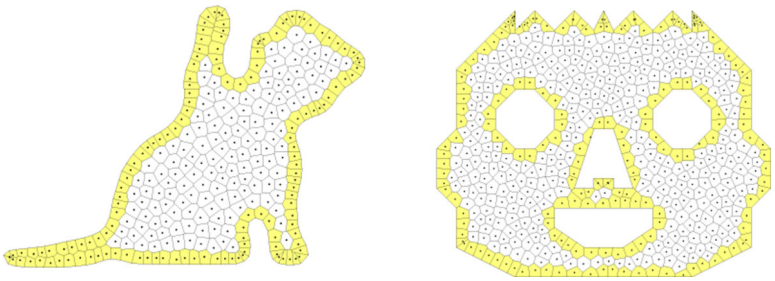
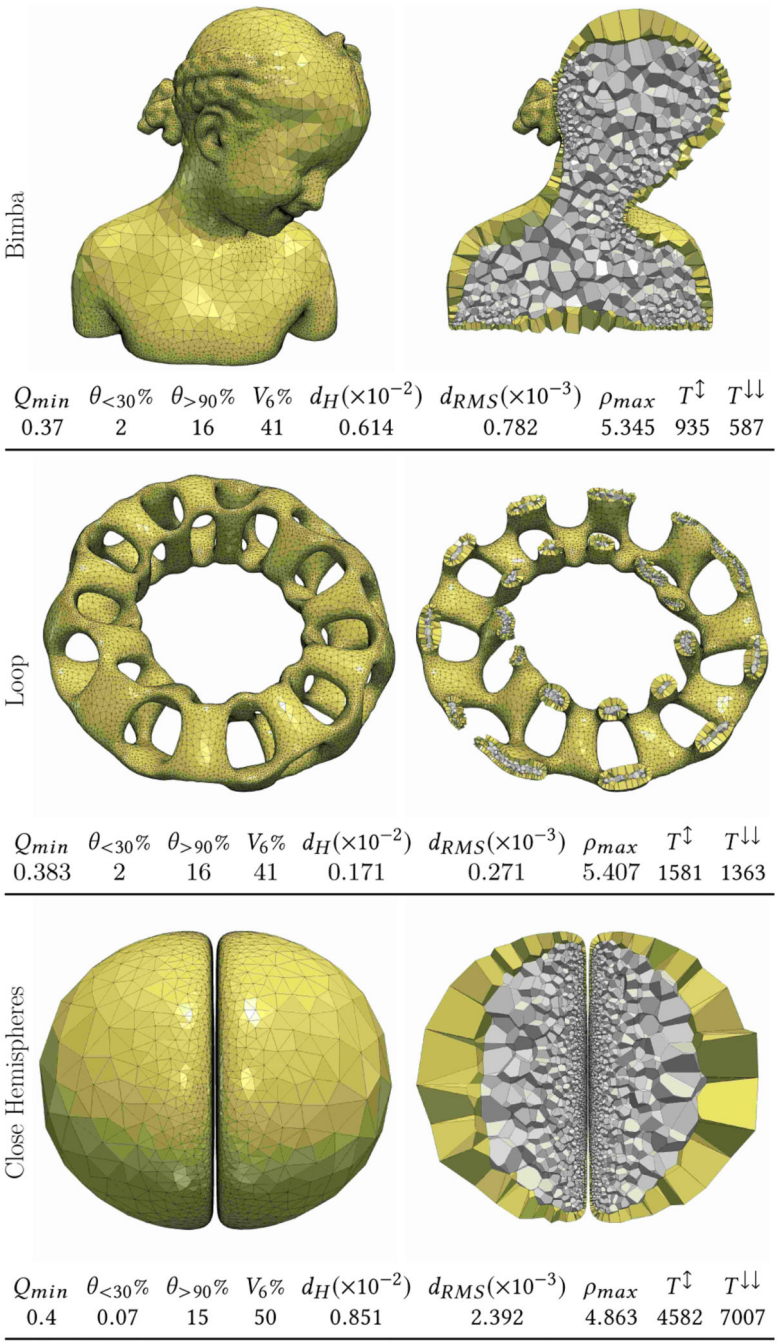


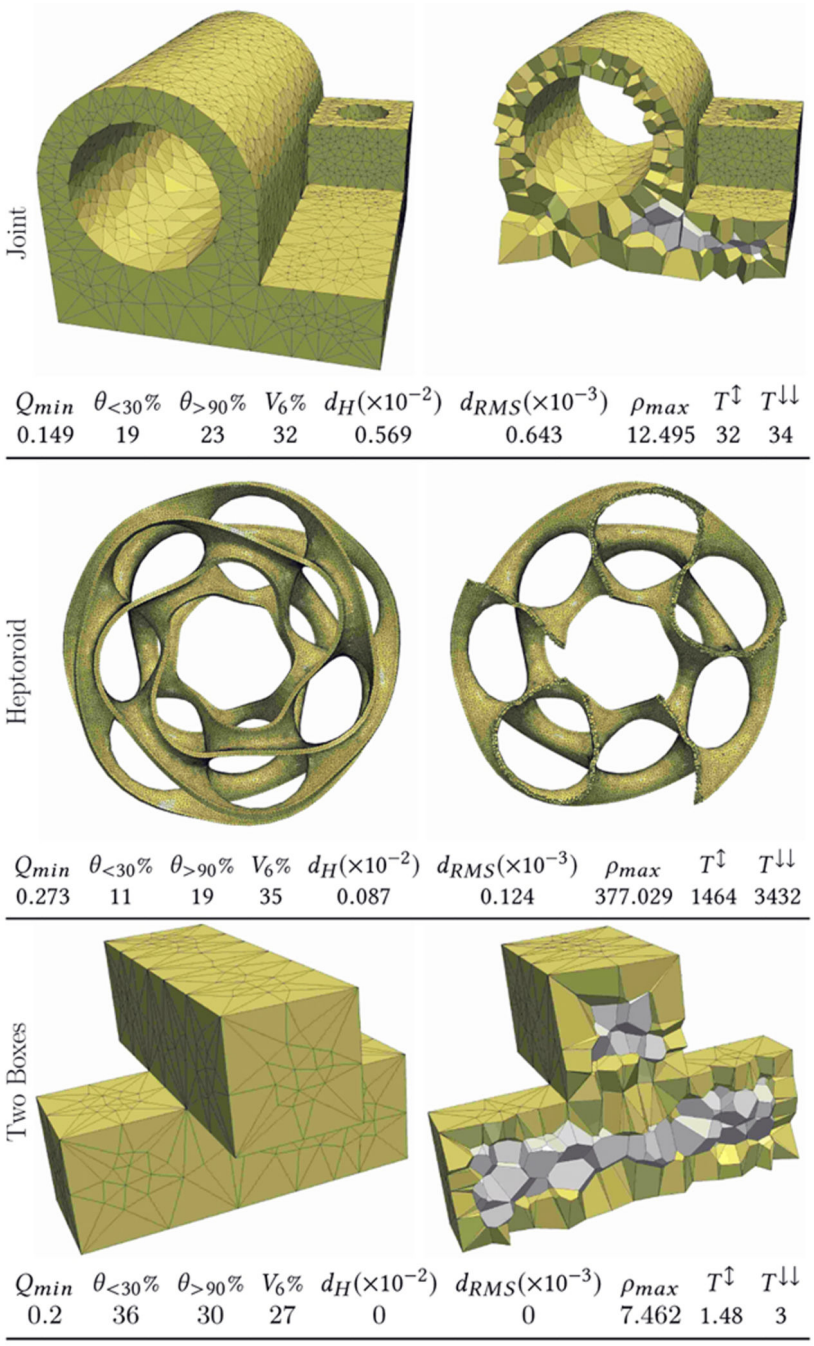
Fig. 10.
The VoroCrust algorithm readily handles 2D domains.



Q_{min}	$\theta_{<30}\%$	$\theta_{>90}\%$	$V_6\%$	$d_H(\times10^{-2})$	$d_{RMS}(\times10^{-3})$	ρ_{max}	$T^{\updownarrow}$	$T^{\downdownarrows}$
0.37	2	16	41	0.614	0.782	5.345	935	587
0.383	2	16	41	0.171	0.271	5.407	1581	1363
0.4	0.07	15	50	0.851	2.392	4.863	4582	7007

Fig. 11.

Sample results on smooth models exhibiting detailed features with a large range of feature sizes (top), complex topologies with multiple holes and narrow regions (middle), and multiple components nearly in contact (bottom).



Q_{min}	$\theta_{<30}\%$	$\theta_{>90}\%$	$V_6\%$	$d_H(\times10^{-2})$	$d_{RMS}(\times10^{-3})$	ρ_{max}	$T^{\updownarrow}$	$T^{\downdownarrows}$
0.149	19	23	32	0.569	0.643	12.495	32	34
0.273	11	19	35	0.087	0.124	377.029	1464	3432
0.2	36	30	27	0	0	7.462	1.48	3

Fig. 12.

Sample results on models with sharp features including: mechanical models (top), complex topologies and narrow regions (middle), and non-manifold boundaries (bottom).

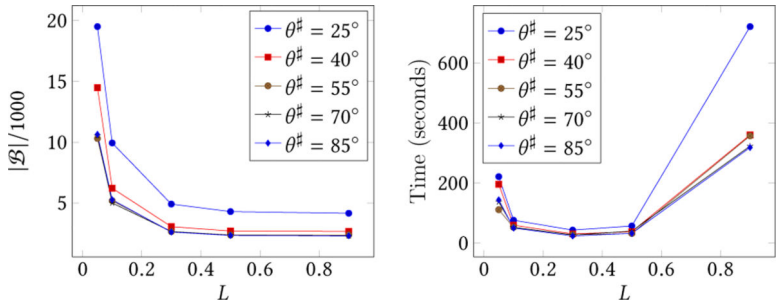


Fig. 13.

Impact of the parameter L on the Joint model for varying values of $\theta^\sharp$. While the level of refinement is inversely proportional to L, increasing L slows down the algorithm due to larger ball neighborhoods.

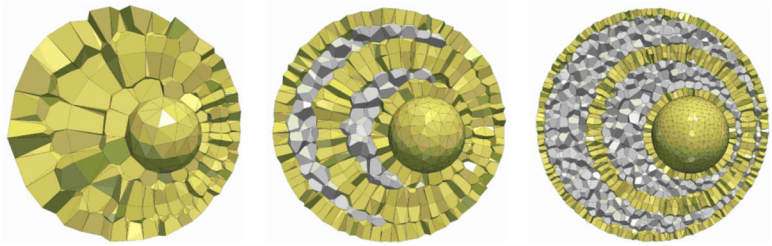
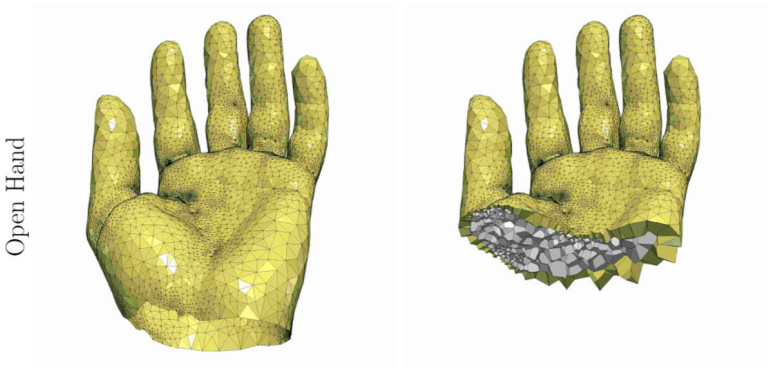


Fig. 14.

Impact of the sizing field parameter *sz* on the nested spheres model. From left to right: *sz* = ∞ (default), *sz* = 1, and *sz* = 0.5.



Q_{min}	$\theta_{<30}\%$	$\theta_{>90}\%$	$V_6\%$	$d_H(\times 10^{-2})$	$d_{RMS}(\times 10^{-3})$	ρ_{max}	$T^{\updownarrow}$	$T^{\downdownarrows}$
0.13	8	18	39	0.884	1.42	16.598	105	151

Fig. 15.
VoroCrust can handle surfaces with boundary.

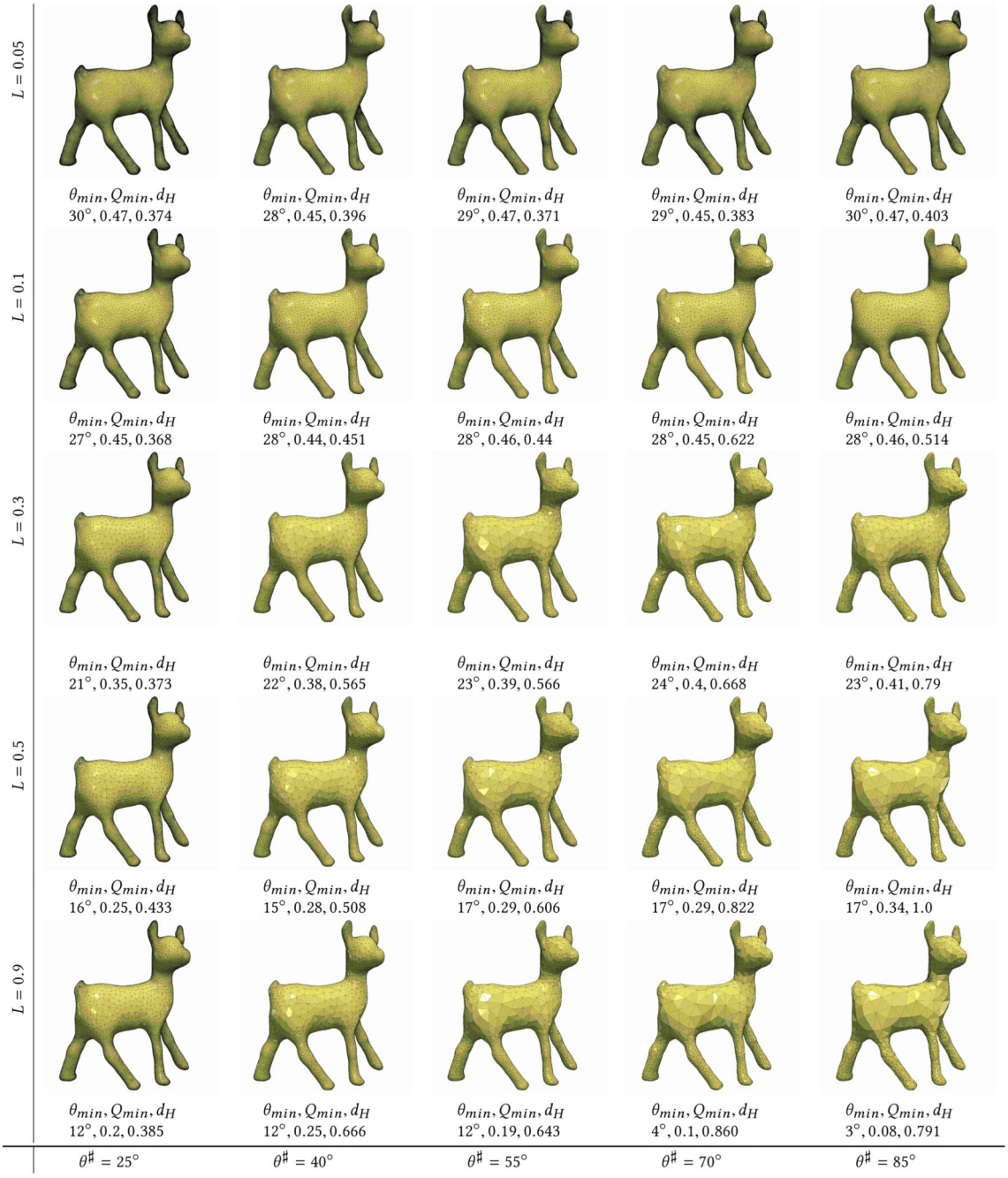


Fig. 16.

The Impact of input parameters L and $\theta^\sharp$ on the surface quality and approximation error, demonstrated on the smooth Goat model. Besides the visual comparison, we report the minimum angle in the surface mesh θ_{min}, the minimum triangle quality Q_{min}, and the Hausdorff error $d_H (\times 10^{-2})$ normalized by the diagonal of the bounding box.