Low-Resolution Remeshing Using the Localized Restricted Voronoi Diagram

Dong-Ming Yan, Guanbo Bao, Xiaopeng Zhang, and Peter Wonka

Abstract—A big problem in triangular remeshing is to generate meshes when the triangle size approaches the feature size in the mesh. The main obstacle for *Centroidal Voronoi Tessellation* (CVT)-based remeshing is to compute a suitable Voronoi diagram. In this paper, we introduce the localized restricted Voronoi diagram (LRVD) on mesh surfaces. The LRVD is an extension of the restricted Voronoi diagram (RVD), but it addresses the problem that the RVD can contain Voronoi regions that consist of multiple disjoint surface patches. Our definition ensures that each Voronoi cell in the LRVD is a single connected region. We show that the LRVD is a useful extension to improve several existing mesh-processing techniques, most importantly surface remeshing with a low number of vertices. While the LRVD and RVD are identical in most simple configurations, the LRVD is essential when sampling a mesh with a small number of points and for sampling surface areas that are in close proximity to other surface areas, e.g., nearby sheets. To compute the LRVD, we combine local discrete clustering with a global exact computation.

Index Terms—Localized restricted Voronoi diagram, centroidal Voronoi tessellation, Poisson-disk sampling, remeshing

1 Introduction

The Voronoi diagram is a fundamental geometric structure that has numerous applications in areas such as mesh generation, physical simulation, visualization, image processing, biology, chemistry, geography and architectural design.

The Voronoi diagram on surfaces is very useful in mesh processing algorithms that distribute sample points on a surface, e.g., algorithms for sampling, remeshing, and rendering. A typical example is remeshing algorithms based on the *Centroidal Voronoi Tessellation* (CVT) [1] because they require computing the Voronoi diagram on the input surface during each iteration.

There are multiple ways to define a Voronoi diagram on surfaces. We analyze possible choices with the goal of low-resolution remeshing in mind. The most important design choice is whether to use geodesic or Euclidean distances for the definition of the Voronoi diagram. Voronoi diagrams using geodesic distances have multiple advantages [2], [3]. They are intuitive to understand, consist of connected Voronoi cells, and are useful for applications such as surface segmentation [4] and base domain construction for multi-resolution modeling [5]. On the downside, the computation time for exact geodesic distances is high. Therefore, several authors propose using approximate Voronoi diagrams by discrete clustering algorithms, e.g., [6]. This results in a low geometric quality of the boundary approximation of the Voronoi cells (see Fig. 1a). Further, even exact geodesic Voronoi diagrams (GVD) are not that suitable for remeshing. For low-resolution remeshing, the geodesic Voronoi diagram is very sensitive to noise and leads to bad triangle angles. The alternative strategy is to compute a three-dimensional (3D) Voronoi diagram in Euclidean space and intersect it with the surface [7], [8]. The advantages of this approach are that fast and exact algorithms are available. A major disadvantage is the problem that Voronoi cells can consist of disconnected regions. This is an obstacle in remeshing of nearby sheet structures with two closeby surface regions, because the points on one side of the nearby sheets create disconnected Voronoi regions on the other side (see Fig. 1b). While the suitability of geodesic versus Euclidean distances for remeshing is an ongoing research question, the current state of the art using Euclidean distances is superior in terms of element quality, scalability, and surface approximation. We therefore aim at designing a Voronoi diagram that is mainly based on Euclidean distances, but redefines the surface boundaries using a more geodesic concept of connectivity so that disconnected regions cannot occur (see Fig. 1c).

In this paper, we propose a new type of Voronoi diagram on surfaces, called the localized restricted Voronoi diagram (LRVD). We provide a definition of the LRVD, and an algorithm to compute the LRVD on mesh surfaces, and we explain how the LRVD can be integrated in to state-of-the-art remeshing algorithms using CVT-based isotropic remeshing and remeshing using maximal Poisson-disk sampling (MPS). In practice, this enables us to overcome two problems: remeshing nearby sheet structures and remeshing with a lower number of vertices. The main contributions of this paper are:

- *D.-M. Yan is with the King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia, and the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China. E-mail: yandongming@gmail.com.*
- *G. Bao is with the Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China. E-mail: guanbo.bao@gmail.com.*
- *X. Zhang is with the National Laboratory of Pattern Recognition-LIAMA, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China. E-mail: xiaopeng.zhang@ia.ac.cn.*
- *P. Wonka is with the King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia, and Arizona State University, Tempe, AZ 85287-8809. E-mail: pwonka@gmail.com.*

Manuscript received 8 Dec. 2013; revised 19 May 2014; accepted 28 May 2014. Date of publication 11 June 2014; date of current version 27 Aug. 2014. Recommended for acceptance by A. Sheffer.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2014.2330574

1077-2626 © 2014 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

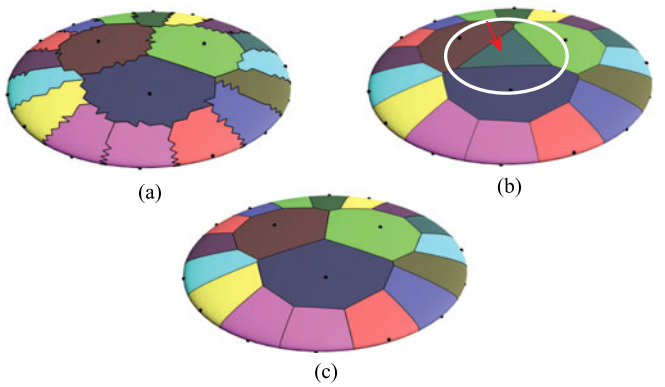


Fig. 1. (a) An approximate Voronoi Diagram on a surface computed using discrete clustering. The approximation results in jagged boundaries. (b) An exact RVD computed by intersecting a 3D Voronoi diagram and the input mesh. Note how a point on the backside of the model generates a disconnected Voronoi region (highlighted by the white circle). (c) The proposed LRVD has no disconnected regions and high-quality geometric boundaries.

- The introduction of a new type of Voronoi diagram on surfaces, the LRVD.
- An algorithm for computing the LRVD that is more efficient than the RVD algorithm.
- The application of the LRVD concept to improving the geometric quality and computation speed of two state-of-the-art remeshing algorithms.

1.1 Related Work

We briefly review the literature of Voronoi diagrams on surfaces and their applications. For more details, we refer readers to recent surveys [4], [9] and textbooks [10], [11].

The most natural way to define the Voronoi diagram on surfaces uses the geodesic metric, resulting in the geodesic Voronoi diagram. Kunze et al. [2] propose a method for computing the GVD on parametric surfaces by tracing isovalue curves in the parameter domain. Peyré and Cohen [12] propose an approximated approach to computing the GVD using the fast marching algorithm. Liu et al. [3] present an exact algorithm to compute isocontours, bisectors and GVDs on mesh surfaces. Although the exact GVD has many nice properties, the computational cost is too expensive. Additionally, for remeshing applications, the geodesic distance is not the best choice since the remeshing quality is still measured in the Euclidean space.

An alternative way to approximate the GVD is to replace the geodesic metric with the Euclidean metric. Edelsbrunner and Shah [13] define the restricted Voronoi diagram (RVD) on surfaces as the intersection of the 3D Voronoi diagram and the surface. Du et al. [14] show that when the sampling set is dense enough, the Euclidean distance is a good approximation of the geodesic distance. Later, several approximation algorithms were proposed for computation of the RVD. Alliez et al. first sample the input surface with enough quadrature samples, and then apply k-means clustering with quadratures to approximate the RVD on the surface [15]. Valette and Chassery, [6], [16] propose the use of mesh triangles as the basic primitive for clustering. The result of the discrete clustering is then used to approximate the RVD. The advantages of the approximated RVD include simple implementation and fast computation. However, the subsequent mesh-processing algorithms can suffer due to inexact computation.

Yan et al. [7] propose an efficient implementation for the exact computation of the RVD that first builds a kd-tree from the samples, and then finds the nearest sample point for each mesh triangle. The incident Voronoi cells of a triangle can be efficiently identified by traversing the neighboring cells of the current cell. The algorithm complexity is $O(m\,log(n))$. The RVD computation technique has been used in multiple algorithms, such as isotropic remeshing [7], anisotropic/quad remeshing [8], quadrilateral surface fitting [17], minimal surface modeling [18], blue noise sampling on surfaces [19], and function approximation on surfaces [20]. Yan and Wonka [21], [22] generalize the restricted Voronoi diagram to the restricted power diagram (RPD) on surfaces. Sun et al. [23] apply a hexagonal metric to compute an approximate anisotropic RVD on surfaces.

On the theoretical side, Leibon and Letscher [24] first show that a valid intrinsic Delaunay triangulation on a manifold can be ensured by the sampling density only. The recent work of Boissonnat et al. [25] shows that the result of [24] is incorrect, because the topology of the intrinsic Delaunay triangulation on a manifold can also be invalid if the vertices lie too close to a degenerate or a "quasi-cospherical" configuration.

2 Definitions

In this section, we first review the definitions of the restricted Voronoi diagram and its dual, the restricted Delaunay triangulation (RDT). Then, we introduce our generalization to the localized restricted Voronoi diagram and the localized restricted Delaunay triangulation (LRDT).

2.1 Restricted Voronoi Diagram

Given a smooth two-manifold surface $S \in \mathbb{R}^3$ and a set of finite samples $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ on S, the restricted Voronoi diagram is defined as the intersection of the 3D Voronoi diagram $\Omega = \{\Omega_i\}_{i=1}^n$ of $\mathbf{X}$ and the surface S [13] (see Fig. 2a for a two-dimensional (2D) illustration of the RVD):

$$\Omega_{|S} = \Omega \cap S = \{\Omega_i \cap S\}_{i=1}^n,$$

where

$$\Omega_i = \{\mathbf{x} \in \mathbb{R}^3, d(\mathbf{x}, \mathbf{x}_i) \leq d(\mathbf{x}, \mathbf{x}_j), \forall \mathbf{x}_j \in \mathbf{X}, j \neq i\}.$$

Then, a restricted Voronoi cell (RVC) is defined as:

$$\Omega_{i|S} = \{\mathbf{x} \in S, d(\mathbf{x}, \mathbf{x}_i) \leq d(\mathbf{x}, \mathbf{x}_j), \forall \mathbf{x}_j \in \mathbf{X}, j \neq i\}.$$

In the above formula, $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$ is the Euclidean distance between two points.

2.2 Restricted Delaunay Triangulation

The dual of the RVD is a subcomplex of the 3D Delaunay triangulation, called the restricted Delaunay triangulation. If the ϵ-sampling property [26] and the topological ball property [13] are met, each restricted Voronoi cell is a single connected component and the RDT is topologically equivalent to the underlying surface S.

The RVD and the RDT work well for general smooth surfaces when certain properties are met. However, if the surface has nearby sheets, or if different parts of the surface are too close to each other (e.g., self-intersecting surfaces), the

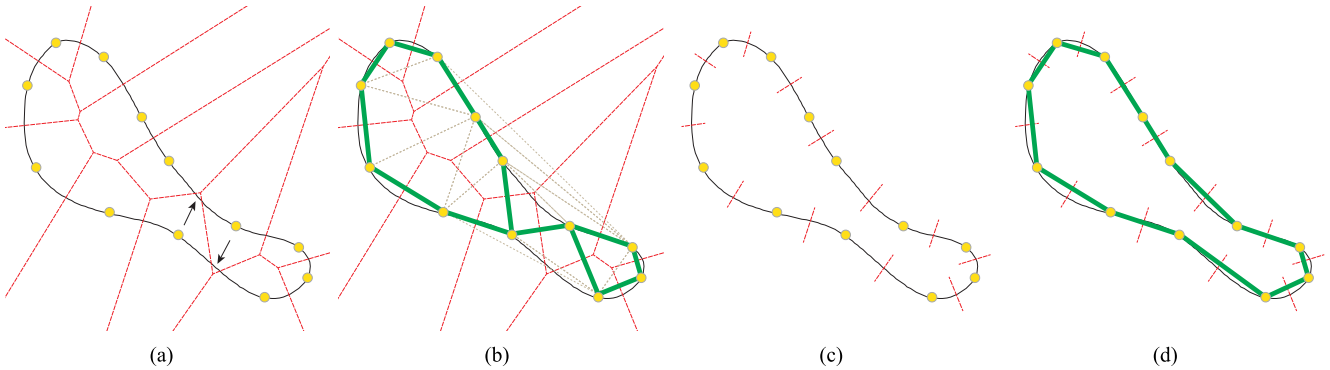


Fig. 2. Two-dimensional illustration of RVD/RDT and LRVD/LRDT. The yellow dots are sample points on the curve. (a) The RVD has disconnected RVCs in nearby sheet regions if the sampling density does not meet the ϵ-sampling property (see the black arrows). (b) The dual RDT is not homeomorphic to the input curve. (c) LRVD of our method and (d) a valid LRDT.

RVD and RDT might not be valid any more. In such a case, a cell of the RVD might have multiple connected components, which makes the topology of the dual RDT inconsistent with the surface S. A 2D example of this problem is illustrated in Fig. 2b.

2.3 The Localized Restricted Voronoi Diagram

To address the topological inconsistency problem caused by the RVD, we introduce the concept of the localized restricted Voronoi diagram. The LRVD is defined in an intrinsic manner by using the Euclidean metric instead of using the geodesic metric.

Given an input surface S and a set of points $\{\mathbf{x}_i\}_{i=1}^n$ sampled on S, the LRVD $\Omega'_{|S}$ is defined as a tuple of localized restricted Voronoi cells (LRVCs) $\{\Omega'_{i|S}\}_{i=1}^n$, such that the following properties are fulfilled:

- The generating point $\mathbf{x}_i$ is inside $\Omega'_{i|S}$ for each $\Omega'_{i|S}$.
- Each $\{\Omega'_{i|S}\}$ is connected.
- At the boundary of a pair of neighboring cells $\Omega'_{i|S}$ and $\Omega'_{j|S}$, the Euclidean distance between the generating points $\mathbf{x}_i$ and $\mathbf{x}_j$ to any boundary point has to be identical.
- Two cells can intersect only at their boundaries.
- The union of the cells covers the input surface, i.e., $\bigcup_{i=1..n}\Omega'_{i|S} = S$.

In simple cases, the LRVD is equal to the RVD (if there are no disconnected cells in the RVD). Otherwise, the LRVD can be unique (if there is only one way to redistribute parts of the disconnected cells from the RVD to its neighboring cells). In complex cases, multiple LRVDs may exist, e.g., if the number of samples is too small for a surface with a complicated geometry, each sample might generate multiple disconnected RVCs. In such cases, there are multiple ways to reassign the disconnected cells. However, in the next section, we present a practical algorithm for computing a unique LRVD that fulfills the above properties.

The dual of the LRVD is called the localized restricted Delaunay triangulation. A 2D example of the LRVD/LRDT is illustrated in Figs. 2c and 2d. We refer readers to [11] and [25] for a theoretical analysis of the topological validity of the intrinsic Delaunay triangulation on a manifold. Note that the word "localized" has also been used in [27] for other purposes.

2.4 Extension to the Power Diagram

If we change the Euclidean distance with the power distance $d_P(\mathbf{x}, w_x, \mathbf{y}, w_y) = \|\mathbf{x}-\mathbf{y}\|^2 - w_x - w_y$ (where w_x, w_y are the weights of points $\mathbf{x}, \mathbf{y}$, respectively), then the concepts of the RVD and the RDT can be generalized to the restricted power diagram and the restricted regular triangulation (RRT), respectively [21]. Similarly, the LRVD and LRDT can also be generalized to the localized restricted power diagram (LRPD) and the localized restricted regular triangulation (LRRT), respectively. In the following, we describe the computation of the LRVD, because the algorithm for the LRPD is just a simple variation.

3 LRVD Computation

In this section, we introduce an algorithm for computing the LRVD. The inputs of our algorithm are a surface S and a set of sample points $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ on the surface. S is assumed to be a two-manifold triangular mesh, with or without boundaries, that consists of a set of triangles $T = \{t_j\}_{j=1}^m$. The output of the LRVD computation is a collection of LRVCs $\{\Omega'_{i|S}\}_{i=1}^n$. Each LRVC $\Omega'_{i|S}$ is a collection of connected polygons. Each polygon is either a triangle of the input mesh or part of a triangle that is clipped by bisecting planes between $\mathbf{x}_i$ and the neighboring samples of $\mathbf{x}_i$ that are incident to the triangle. In order to simplify the algorithm description, we assume that the number of triangles is much larger than the number of samples, i.e., $m \gg n$, and that each triangle contains at most one sample point. Later, we describe why the proposed algorithm works well for general cases in practice.

Recall that the RVD is defined as the intersection of the 3D Voronoi diagram and the mesh surface. The key problem is to identify the incident Voronoi cells for each mesh triangle and compute their intersection. We say that a triangle is incident to a Voronoi cell if a triangle is partially contained in the Voronoi cell. Fig. 3 shows an example of the RVD and the RDT on a mesh surface. There are three types of vertices in the RVD: (a) the original mesh vertices; (b) the intersections of a bisecting plane and a mesh edge, and (c) the intersections of a Voronoi edge and a mesh triangle. Each type (c) vertex of the RVD is dual to a triangle in the RDT. The similarity between the RVD and LRVD is that the cell boundaries are computed as intersections between mesh triangles and bisecting planes. The difference between RVD and LRVD is that the number of sample points that

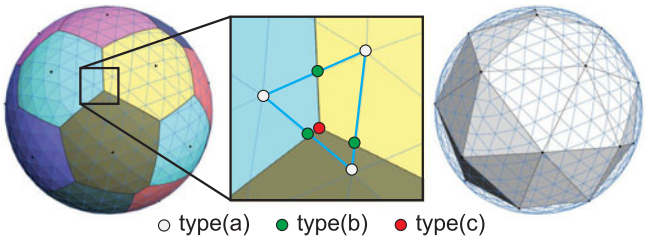


Fig. 3. An example of the RVD (left), a zoom-in (middle), and the RDT (right) on a sphere.

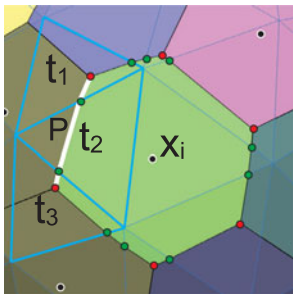


Fig. 4. Illustration of the structure of RVCs. The boundary segments consist of intersections of bisecting planes and mesh triangles.

can contribute bisecting planes is limited locally to eliminate disconnected cells. Therefore, both the RVD and LRVD computations need to find the incident cells for each input triangle t_j and clip the triangle with respect to the bisecting planes. However, the main difference is that we need to compute the local validity of cells via a region-growing approach instead of via the global intersection.

3.1 Algorithm Overview

We use the following key observations about the structure of the LRVD in our algorithm. The outer boundary of a valid LRVC consists of one or multiple closed loops that consist of a set of simple connected segments. The end points of the segments are type (b) or (c) vertices as shown in Fig. 3.

Each segment is the intersection of a bisecting plane and a mesh triangle. The left side of Fig. 4 shows such an example. When a boundary loop of the LRVC crosses a type (c) vertex, it changes the underlying bisecting plane but still lies inside the same mesh triangle. When a boundary loop crosses a type (b) vertex, it changes the containing mesh triangles but keeps the same bisection plane. This case helps us to design a simple test of the local correctness of the LRVD. If two triangles share a type (b) vertex on a common

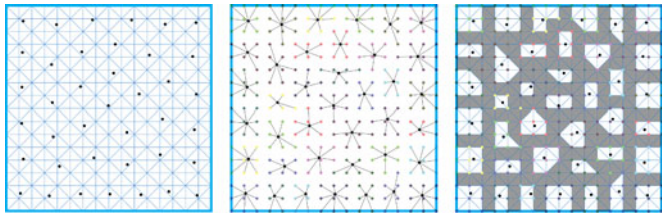


Fig. 5. Given an input mesh and a set of samples (left), the sample points are first assigned to a cell (middle). Grouping of triangles (right): triangles with all vertices assigned to the same region are white; triangles where the vertices are assigned to different regions are dark gray.

edge, then these two neighboring triangles are incident to the same two Voronoi cells. Note that an RVC can be bounded by multiple loops (see Fig. 13 for examples).

Based on the observations above, we propose a three-step algorithm for the LRVD computation. In the first step, we initialize the cells by assigning a cell label to each mesh vertex. In the second step, we assign a list of cell labels to each triangle. The triangles with more than one label assigned are set as *active*. In the third step, the active triangles are clipped by the incident bisecting planes. Then, the local boundary configuration is tested. If found invalid, the boundary is propagated to neighboring triangles. The third step is repeated until there remains no active triangle (as shown in Fig. 6). We detail these steps next.

3.2 Initial Cell-to-Vertex Assignment

The goal of this step is to construct approximate Voronoi cells by assigning an initial Voronoi cell label to each mesh vertex. This assignment is approximate. We follow the assignment algorithm from Cohen-Steiner et al. [28] by replacing the $L^{2,1}$ metric with the L^2 metric, which preserves the connectivity of each Voronoi cell. Figs. 5a and 5b illustrate this algorithm step.

A priority queue Q is used for this assignment step. Q is initialized by an initial set of vertex-cell pairs $(\mathbf{v}_j, \mathbf{x}_i)$. The Euclidean distance $\|\mathbf{v}_j, \mathbf{x}_i\|$ is used as the priority of the queue elements. Smaller distances have higher priority. Additionally, each vertex has a flag *assigned* (initialized with false) that indicates if the assignment is finished.

We first traverse the input triangles that contain a sample point $\mathbf{x}i$. For each such triangle, we create three vertex-cell pairs for its vertices and push them into Q. For this assignment, the cell induced by $\mathbf{x}i$ is used, even though that might not be the closest sample point to a vertex. After initialization, the highest priority vertex-cell pair from Q is processed

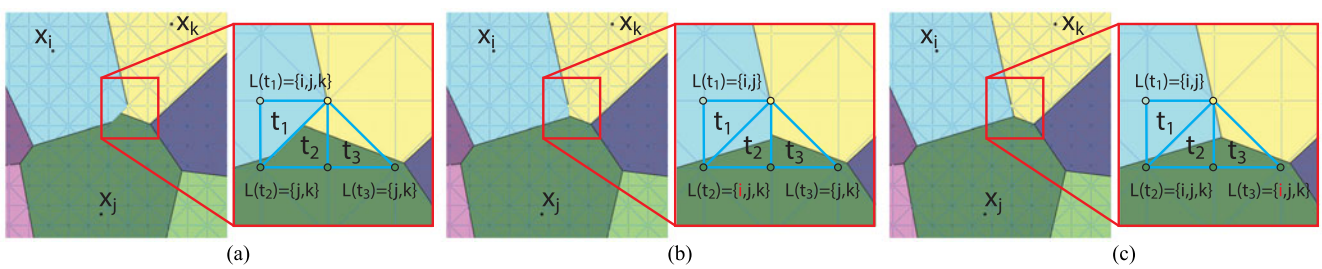


Fig. 6. A 2D Illustration of LRVD computation. (a) The initial active triangle list $\{t_1, t_2, t_2\}$ and the initial incident cell lists are $L(t_1) = \{i, j, k\}, L(t_2) = \{j, k\}, L(t_3) = \{j, k\}$; (b) after one step propagation, cell i is propagated from t_1 to t_2 and is added to the incident cell list of t_2, such that $L(t_2) = \{i, j, k\}$; (c) after two iterations, cell i is added to t_3's incident cell list, and $L(t_3) = \{i, j, k\}$.

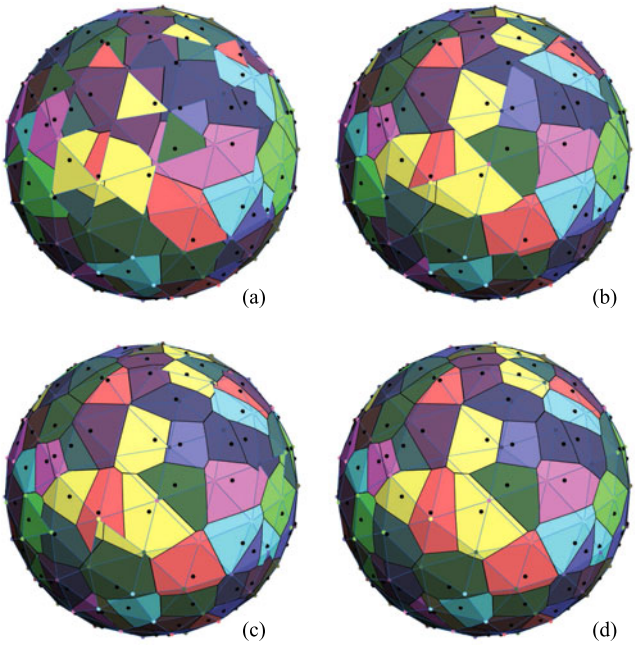


Fig. 7. LRVD computation on a sphere. (a)-(d) show the effect of the local propagation algorithm.

until Q is empty. Processing a pair $(\mathbf{v}_j, \mathbf{x}_i)$ consists of the following steps: we first check whether or not $\mathbf{v}_j$ is assigned to a cell. If not, the vertex $\mathbf{v}_j$ is assigned to the cell of $\mathbf{x}_i$ and the assignment flag is set to true. Then, we traverse the neighboring vertices $\{\mathbf{v}_{j,k}\}$ of $\mathbf{v}_j$. If a neighboring vertex $\mathbf{v}_{j,k}$ is not assigned with any cell, then we push the new pair $(\mathbf{v}_{j,k}, \mathbf{x}_i)$ into Q. This process is repeated until Q is empty.

3.3 Initial Cell-to-Triangle Assignment

The goal of this step is to assign an initial set of cells to each mesh triangle t_j and to build a list of active triangles. We maintain a list L_{t_j} to store the indices of potentially incident cells for each active triangle t_j. We first traverse each input triangle t_j and add the cell indices of three vertices of t_j to the list L_{t_j}. Note that there is a special case that we have to consider separately: a triangle t contains a sample point $\mathbf{x}_i$, but none of t's vertices is assigned to the ith cell. In this case, we also add i to t's list L_t. We then set to be active all triangles that have more than one cell assigned to them. At the end of this step, each triangle is assigned one to four cells that are potentially active. This is correct if the assumption that each triangle can contain at most one sample point holds.

3.4 Local Propagation

First, each triangle t_i is split into $|L_{t_i}|$ convex polygons $\{C(t_i, k)\}$. Each polygon $C(t_i, k)$ is associated with the corresponding LRVC of sample $\mathbf{x}_k$ and describes the part of the triangle that should be assigned to cell k. Since the list L_{t_i} is only approximate, the splitting algorithm is designed to result in empty polygons for all cells $\in L_{t_i}$ that are not incident to the triangle. The splitting algorithm works as follows. For each cell $k \in L_{t_i}$ the polygon is computed by clipping the triangle with all bisecting planes between $\mathbf{x}_k$ and all $\mathbf{x}_l$ with $l \in L_{t_i}, l \neq k$.

Fig. 6 (left) shows a simple example. Next, we check the consistency of the bisecting plane between each pair of neighboring triangles. The convex polygons are again defined as type (a), (b), and (c) vertices as defined before. There is a bisecting plane generating each type (b) vertex (a vertex on an edge of the mesh triangle). We check that the two cell indices generating each such bisecting plane are also present in the cell list of the neighboring triangle and mark the triangle as active. This step is repeated until no active triangle remains. This algorithm always converges in a finite number of steps. Fig. 7 shows the LRVD computation process on a sphere.

3.5 Extensions

Here, we briefly discuss extensions and specialized cases. If a triangle t_j contains more than one sample point, the algorithm can simply be extended by adding all of these sample points to the list L_{t_j} in step 2 of the algorithm. To extend the algorithm to power diagram computation, we simply change the Euclidean distance to the power distance. The power diagram requires each sample point to have a weight (equivalently where the weight is the square of the sampling radius).

3.6 Implementation

We use the symbolic representation for RVD vertices as provided in [8]. We use exact predicates [29] during the clipping process to avoid numerical issues. This is a standard method to ensure robustness of geometric algorithms and it is also implemented in CGAL [30].

4 Applications

In this section, we present two applications that can benefit from the proposed LRVD concept, centroidal Voronoi tessellation and maximal Poisson-disk sampling.

4.1 Centroidal Voronoi Tessellation of Surfaces

Centroidal Voronoi tessellation [1] builds on a special type of Voronoi diagram that requires each generater to coincide with the centroid of its Voronoi cell. CVT attempts to minimize an energy function (Eq. 1) that describes the quantization noise power [31]:

$$F(\mathbf{X}) = \sum_{i=1}^{n} \int_{\Omega_{i|S}} \rho(\mathbf{x}) \|\mathbf{x} - \mathbf{x}_i\|^2 \, dx, \tag{1}$$

where $\rho(\mathbf{x})$ is a density function defined over the surface. The CVT algorithm starts with an initial set of points distributed on the surface and then iteratively moves the points according to the following two steps. 1) Compute a Voronoi diagram of the point set on the surface. The current state of the art uses one of the Voronoi diagram versions described in Section 1.1 with RVD being the current best choice for remeshing quality. 2) Move the points on the surface in order to minimize (Eq. (1)). A modern version of CVT uses a quasi-Newton method [32] that was shown to have better convergence than the original Lloyd relaxation.

The most difficult and time-consuming part in computing a CVT on a mesh surface is the computation of the RVD in each iteration. Our improved CVT algorithm simply

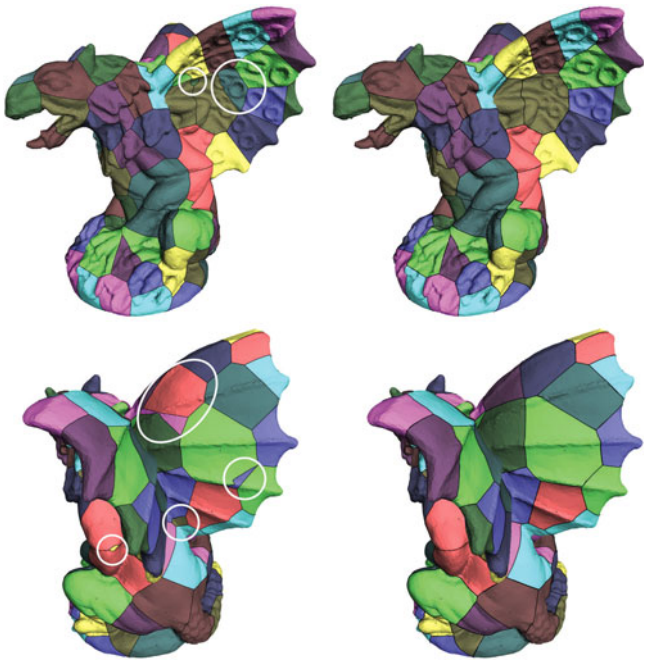


Fig. 8. Comparison of the RVD (left) with the LRVD (right). Thirty points are sampled on the input mesh, and the disconnected components of the RVD are highlighted. More results can be found in the supplemental materials, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2014.2330574.

replaces the RVD with the proposed LRVD. In the results (Section 5), we show how this enables us to handle more complicated inputs, especially models with nearby sheet structures and remeshings with low resolution.

4.2 Maximal Poisson-Disk Sampling on Surfaces

Poisson-disk sampling is a way to generate point-sets with many desirable properties [33]. Given a surface S, a minimal

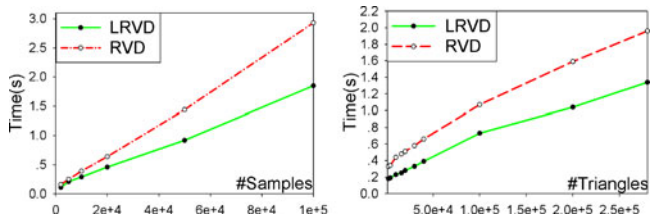


Fig. 9. Timing comparison.

sampling radius r_{min}, a density function $\rho(\mathbf{x})$, and a function $r(\mathbf{x})$ that maps $\rho(\mathbf{x})$ to a sampling radius, a Poisson-disk sampling algorithm randomly generates points on the surface until it is fully covered by disks centered at the points. A typical Poisson-disk set should satisfy three properties: 1) The unbiased sampling property; 2) the minimal distance property; and (3) the maximal sampling property. A sampled point set that satisfies all three properties is called Maximal Poisson-disk sampling.

In this paper, we build on the MPS algorithm of Yan and Wonka [21] that proceeds in the following steps: 1) Subdivide the bounding box of the input mesh into a regular 3D grid. The grid size is derived from the minimal sampling radius. 2) Build an initial point set by performing dart throwing on the surface using the 3D grid to check for conflicts. A dart that is conflicted with an existing point is rejected. 3) After too many consecutive rejections (300 in our implementation) are observed, the algorithm extracts uncovered regions, called gap primitives. The restricted power diagram is used for gap computation. The gap primitives are extracted by subtracting each 3D disk $(\mathbf{x}_i, r_i)$ from the restricted power cell $\Omega_{i|S}$. 4) Perform dart throwing in the extracted gap primitives. 5) Iterate steps 3 and 4 until no more points can be placed.

We propose three modifications to this algorithm. First, in the initialization step 1), we additionally pre-process the input mesh by recursively splitting its long edges. For

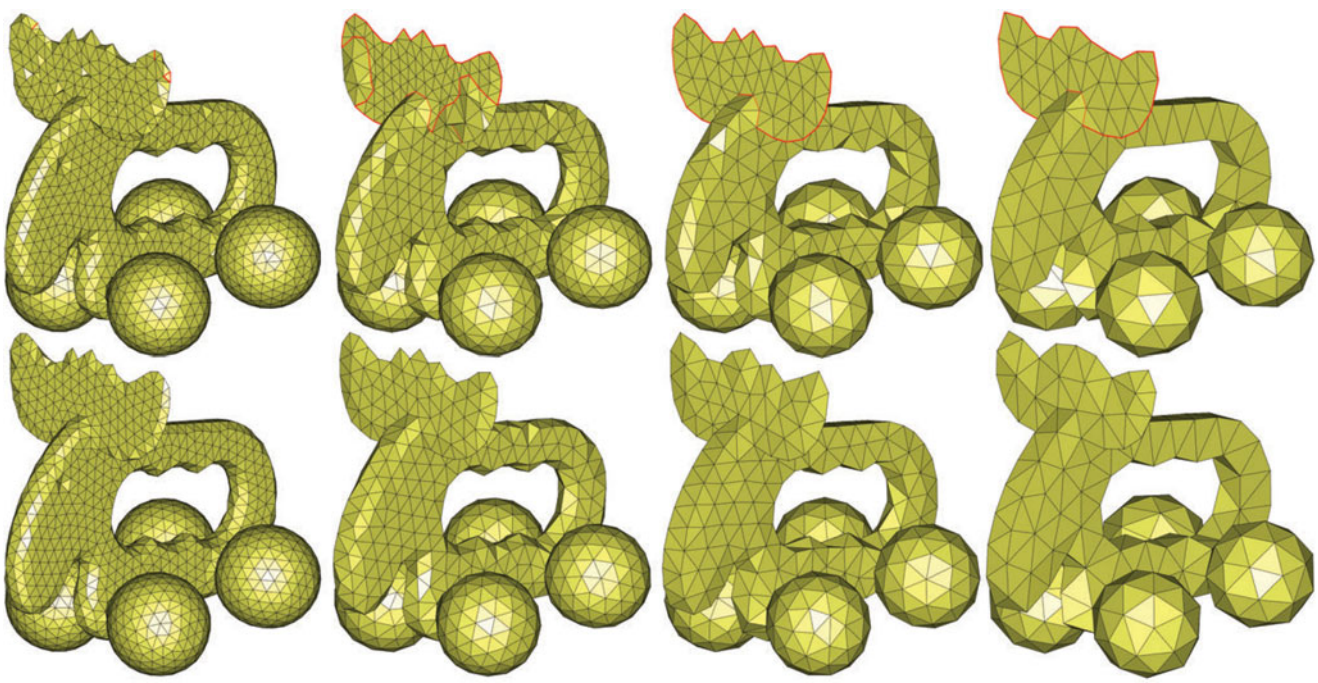


Fig. 10. Comparison of CVT-based remeshing by using RVD (top) and LRVD (bottom). The number of vertices are 3k, 1k, 500, and 300 from left to right. The non-manifold and boundary edges are shown in red. See supplemental materials, available online, for a complete set of results.

TABLE 1
Progressively Remeshing the Elk Model with a Decreasing Number of Vertices

#V	Alg.	Q_{min}	$\theta_{min}/\theta_{max}$	$<30^0$	Hdist
4,000	RVD	0.62	31.6/101.2	0	0.42%
	LRVD	0.61	32.5/100.3	0	0.42%
3,002	RVD	0.50	20.0/112.1	0.25%	0.57%
	LRVD	0.53	24.3/109.3	0.22%	0.52%
2,019	RVD	0.49	20.1/111.9	1.09%	0.68%
	LRVD	0.55	26.1/107.9	0.06%	0.55%
1,051	RVD	0.41	16.9/116.7	2.51%	0.71%
	LRVD	0.57	26.4/103.0	0.40%	0.71%
873	RVD	0.40	16.8/114.5	3.06%	1.16%
	LRVD	0.50	21.8/111.0	0.91%	0.76%
516	RVD	0.32	11.9/102.5	3.06%	1.67%
	LRVD	0.41	16.7/111.9	1.65%	1.08%

All remeshings have a small number of vertices and are therefore very challenging for the current state of the art. We evaluate the meshing quality using several metrics described in the text.

TABLE 2
Remeshing Quality Comparison with Previous Approaches.

Model	Alg.	#V	Q_{min}	$\theta_{min}/\theta_{max}$	$<30^0$	Hdist
Ellp.	RVD	619	0.34	14.6/133.1	19.3%	0.70%
	ACVD	400	0.46	21.4/121.0	0.75%	1.92%
	LRVD	400	**0.66**	**37.1/96.43**	**0**	**0.45%**
Elk	RVD	2.3k	0.23	8.16/131.1	1.24%	0.63%
	ACVD	2.0k	0.10	6.03/167.3	6.83%	0.93%
	LRVD	2.0k	**0.52**	**22.2/108.6**	**0.28%**	**0.62%**
Feline	RVD	4.3k	0.07	2.57/162.2	3.33%	**0.44%**
	ACVD	4.0k	0.39	24.4/128.7	0.51%	0.66%
	LRVD	4.0k	**0.51**	**28.4/113.9**	**0.25%**	0.54%
Knot	RVD	2.2k	0.39	15.7/107.3	0.22%	**0.75%**
	ACVD	2.0k	0.48	28.2/117.9	0.07%	1.09%
	HCVT	2.0k	0.51	31.4/110.8	0	1.04%
	LRVD	2.0k	**0.53**	**34.7/99.75**	**0**	0.87%

#V is the number of vertices in the remeshing. While the RVD method starts out with the same number of vertices as the other methods, new vertices are inserted during the optimization to ensure the topological correctness. The best results are highlighted in bold font.

each mesh edge whose length is longer than r_{min}, the two neighboring triangles are split into four. The splitting step terminates when all the edge lengths are smaller than r_{min}. This ensures that each triangle can contain at most one sample point. Second, in the dart-throwing steps 2) and 4), we modify the conflict check by using the triangles instead of the 3D grid. Each triangle is equipped with a list that stores the indices of the disks that fully cover the triangle. In the dart-throwing step, each time a new dart $(\mathbf{x}_i, r_i)$ is randomly generated in triangle t_j, we first collect a set of neighboring triangles starting from t_j by region growing. Neighboring triangles are recursively added to the set until the disk $(\mathbf{x}_i, r_i)$ does not intersect with a neighboring triangle. Then, we conduct a conflict test between the new dart and all previously sampled disks that cover a triangle in the triangle set. Third, we compte gaps using the LRVD instead of the RPD in step 3) of the algorithm.

5 Experimental Results

We present experimental results of the LRVD algorithm, as well as sampling/remeshing results based on the CVT and the MPS framework, respectively. The results are tested on an Intel X5680 Dual Core 3.33 GHz CPU with 4 GB memory and a 64-bit Windows 7 operating system.

Quality. We first show several examples of the LRVD computation on objects with nearby sheet structures in Fig. 8. We randomly sample 30 points on each input mesh and compute the RVD and LRVD. Our approach generates valid RVCs even when the number of the samples is extremely small, while the RVD generates many disconnected cells. We note that if the sampling density meets the ϵ-sampling property, then the results of the LRVD are the same as those of the RVD.

Efficiency. We perform two tests to compare the efficiency of the LRVD and RVD methods. In the first experiment, we use a sphere with 20 k triangles as input, and gradually increase the number of samples from 2 k to 100 k. The timing comparison of the two methods is shown in Fig. 9 (left). In the second experiment, we fix the number of samples to 10k and gradually increase the resolution of the input mesh from 500 to 140 k triangles. The timing curves are shown in Fig. 9 (right). The new method is nearly two times faster, including the guarantee that the cells are valid.

CVT/Remeshing.We conduct two tests for comparing the remeshing quality. We compare our approach with previous work in terms of the triangle quality $Q(t) = \frac{6}{\sqrt{3}} \frac{|t|}{p(t)\,h(t)}$, (where $|t|$ is the area of t, $p(t)$ is the half-perimeter of t and $h(t)$ is the longest edge length of t [34]), min/max angles $\theta_{min}/\theta_{max}$, the percentage of the angles that are less than 30 degrees ($<30^o$), and the Hausdorff distance (Hdist).

In the first test, we use the same decreasing number of samples to compare with the previous CVT-based approach using the RVD [7]. The purpose of this test is to show that the LRVD is important for remeshing with a smaller number of vertices. We first run RVD-based CVT optimization, with a

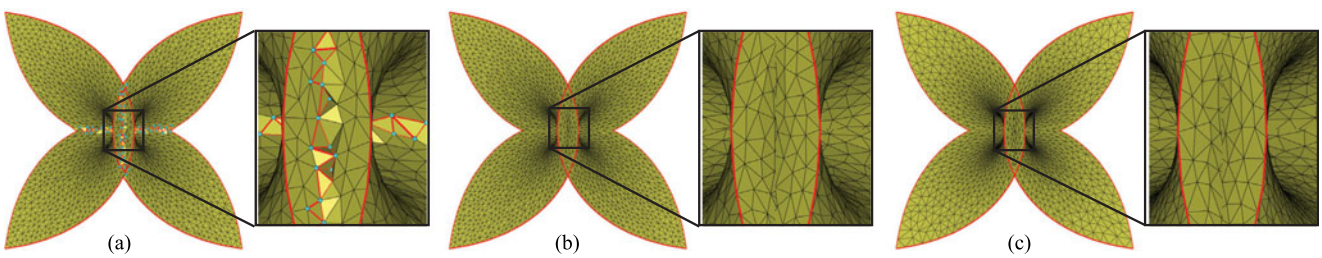


Fig. 11. Maximal Poisson-disk sampling on a self-intersecting surface. (a) Uniform sampling/remeshing of [21]. The red vertices are non-manifold near self-intersection regions. (b) Uniform sampling/remeshing with LRVD. (c) Adaptive sampling/remeshing with LRVD, and right: zoom-in view. The non-manifold vertices are shown in blue and the non-manifold and boundary edges are shown in red.

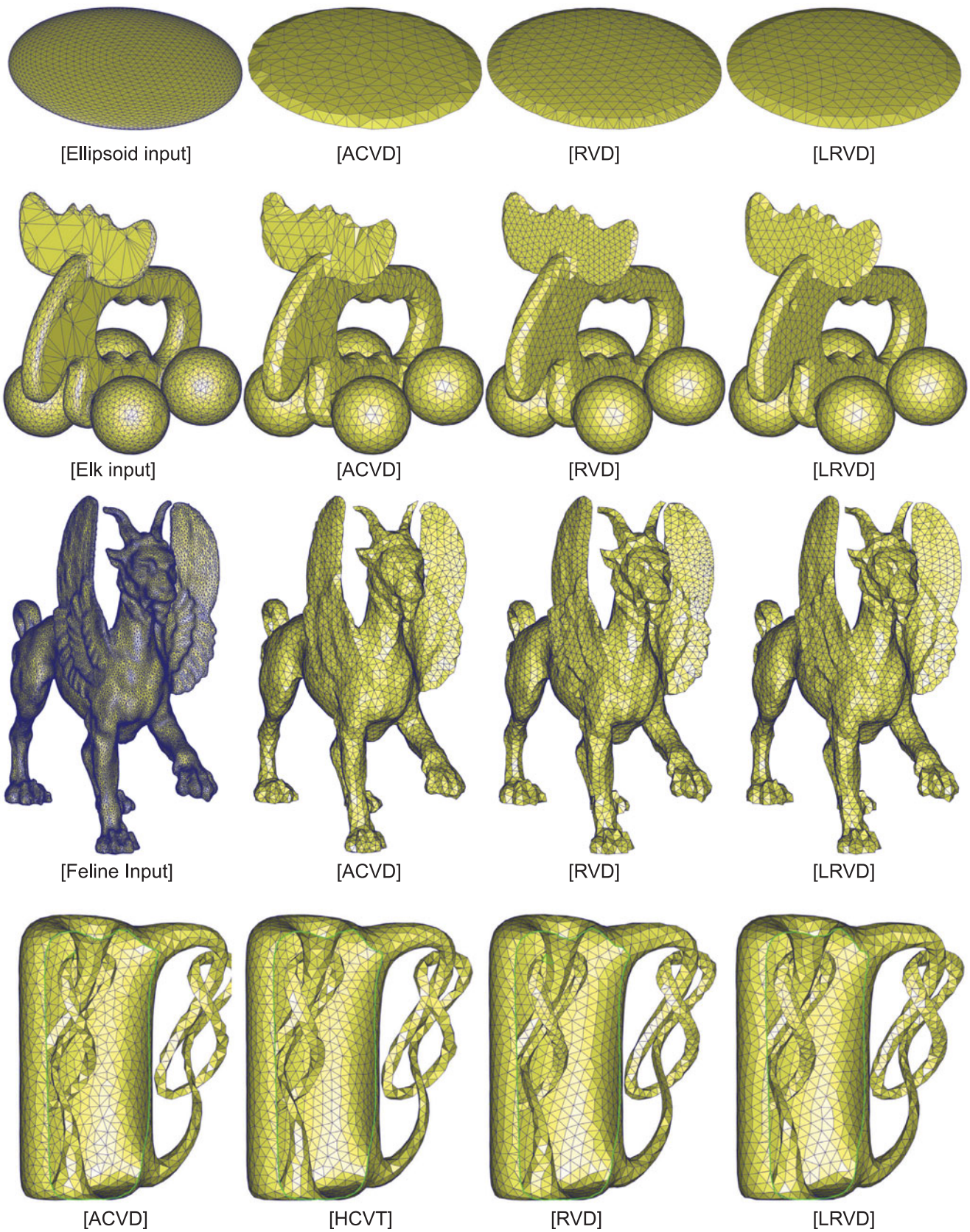


Fig. 12. Comparison of CVT-based remeshing. From top to bottom are flat ellipsoid, Elk, Feline and Knot models. The top three rows show the results of ACVD, RVD and LRVD. The last row shows the cut-view of results generated by ACVD, HCVT, RVD and our result. The quality comparison of these results is given in Table 2.

TABLE 3
MPS Timing Comparison with [21]

Model	r_{min}	Alg.	#V	t_{init}	t_{dt}	t_{mps}	t_{total}
uniform sampling							
Feline	0.005	YW	43.0k	2.2	3.7	15.9	21.8
		Our	42.6k	0.4	2.4	6.9	9.7
VaseLion	0.005	YW	62.8k	2.5	4.9	20.4	27.8
		Our	62.3k	0.6	3.6	9.2	13.4
Dragon	0.005	YW	69.7k	2.9	5.6	19.7	28.2
		Our	69.2k	0.6	4.1	10.2	14.9
Gragoyle	0.005	YW	73.9k	3.5	6.5	20.2	20.2
		Our	72.9k	0.7	4.2	6.9	11.8
adaptive sampling							
Feline	0.004	YW	22.6k	4.2	23.2	10.7	35.1
		Our	22.9k	0.6	6.7	5.5	12.8
VaseLion	0.004	YW	39.6k	4.5	27.2	17.5	49.2
		Our	39.7k	0.9	5.6	6.5	13.0
Dragon	0.004	YW	31.8k	5.2	29.7	13.8	48.7
		Our	31.1k	0.9	6.2	5.7	12.8
Gragoyle	0.004	YW	32.4k	6.5	23.3	12.8	42.2
		Our	32.2k	1.2	6.8	5.3	13.3

decreasing number of sampled points from 4 k to 300. Once the RVD-based CVT optimization converges, we check the disconnected RVCs and insert a new vertex for each connected RVC component as suggested by [8]. Next, we run LRVD-based CVT optimization with the same number of vertices as the output of RVD-based CVT. The Elk model is used for this test. Selected results of both methods are shown in Fig. 10. Results show that the RVD-based approach generates non-manifold geometry in regions containing nearby sheets for all models except for the remeshing with 4 k vertices, while our LRVD-based approach always generates valid results, even for the model with 516 vertices. The comparison of the remeshing quality is given in Table 1. When the RVD-based CVT has enough samples, both methods generate similar results (e.g., 4 k points). However, our meshing quality is clearly better for low-resolution models.

In the second test, we compare several low-resolution remeshing results with three other competing methods. We again compare to Yan et al. [7], but this time we use the original number of vertices that the algorithm started with. This results in Yan et al. [7] having a higher number of vertices than the other methods have. We also compare our approach with the discrete clustering-based method, called *approximated centroidal Voronoi diagram* (ACVD) [6] and a recent parameterization-based approach *hyperbolid centroidal Voronoi tessellation* (HCVT) [35]. This approach can generate a remeshing with a correct topology without inserting new vertices. However, the Hausdorff distance is larger than the others because of the distortion introduced by the parameterization. Fig. 12 shows the comparisons with other methods. The comparisons are listed in Table 2. More comparisons are provided in the supplemental materials, available online.

MPS/remeshing. The MPS algorithm presented in this paper has three advantages compared with the previous approach [21]. First, we do not use a 3D voxel grid and therefore the memory consumption is dramatically reduced. Second, we use the LRPD instead of the restricted power diagram for gap detection and gap filling, which enables us to sample input surfaces with fold-overs or even self-intersections. Fig. 11 shows the results of MPS on a self-intersecting surface. Our algorithm can generate correct results. Third, the computation time is reduced as shown in the following comparisons. We compare the timing of our modified MPS algorithm with [21]. First, we use a sampling radius $r = 0.005$ for uniform sampling (the input models are normalized into the unit cube before sampling). The timings include the initialization, initial dart-throwing and gap-filling for both methods. The comparison shows that the dart-throwing stage of the new method is slower for uniform sampling but faster for adaptive sampling. The overall speed is around two times faster for uniform sampling and three to four times faster for adaptive sampling. The timing comparison is given in Table 3.

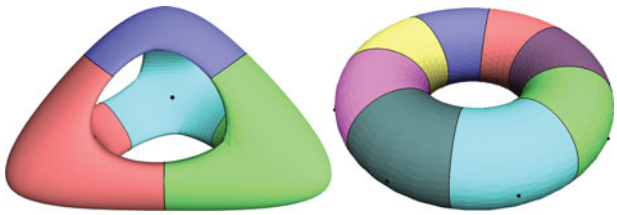


Fig. 13. Failed remeshing. Our remeshing framework cannot generate a valid mesh if the sampling density is too low in tubular regions.

Limitation. One limitation of the LRVD is that the connectivity information of the input mesh is required. We cannot handle triangle soups or noisy data with missing geometry or cracks.

Although the proposed LRVD technique works well for a wide range of inputs, we do not have a theoretical guarantee of the validity of the remeshing. For example, we cannot obtain a valid remeshing if the input contains tubular structures when the number of samples is small. In such a case, more samples are required to satisfy the ϵ-sample property. Fig. 13 shows examples of failed remeshing.

6 Conclusions

We have presented a simple yet efficient approach for computing the localized restricted Voronoi/power diagrams on mesh surfaces. The new approach generates Voronoi cells on surfaces with connected components, which is very useful for CVT-based or MPS-based remeshing. We believe that many other applications can also benefit from this simple technique. In the future, we would like to develop a GPU-implementation to further improve the efficiency .

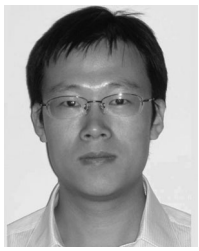
Acknowledgments

The authors thank the anonymous reviewers for their valuable comments and suggestions. They are grateful to Sebastien Valette for the ACVD software and to Xiaohu Guo for the HCVT results. This work was supported by the KAUST Visual Computing Center, the National Natural Science Foundation of China (nos. 61372168, 61331018, 61271431 and 61272327), and the US National Science Foundation (NSF).

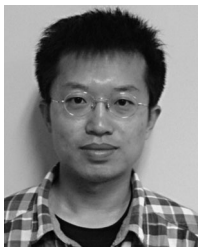
References

[1] Q. Du, V. Faber, and M. Gunzburger, "Centroidal Voronoi tessellations: Applications and algorithms," *SIAM Rev.*, vol. 41, pp. 637–676, 1999.

[2] R. Kunze, F.-E. Wolter, and T. Rausch, "Geodesic Voronoi diagrams on parametric surfaces," in *Proc. Comput. Graph. Int.*, 1997, pp. 230–237.
[3] Y.-J. Liu, Z.-Q. Chen, and K. Tang, "Construction of iso-contours, bisectors and Voronoi diagrams on triangulated surfaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1502–1517, Aug. 2011.
[4] G. Peyré and L. D. Cohen, "Geodesic methods for shape and surface," in *Advances in Computational Vision and Medical Image Processing: Methods and Applications.* New York, NY, USA: Springer, 2008, pp. 29–56.
[5] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle, "Multiresolution analysis of arbitrary meshes," in *Proc. ACM 22nd Annu. Conf. Comput. Graph. Interactive Techn.*, 1995, pp. 173–182.
[6] S. Valette, J.-M. Chassery, and R. Prost, "Generic remeshing of 3D triangular meshes with metric-dependent discrete Voronoi diagrams," *IEEE Trans. Vis. Comput. Graph.*, vol. 14, no. 2, pp. 369–381, Mar./Apr. 2008.
[7] D.-M. Yan, B. Lévy, Y. Liu, F. Sun, and W. Wang, "Isotropic remeshing with fast and exact computation of restricted Voronoi diagram," *Comput. Graph. Forum*, vol. 28, no. 5, pp. 1445–1454, 2009.
[8] B. Lévy and Y. Liu, "L_p centroidal Voronoi tesselation and its applications," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 119:1–119:11, 2010.
[9] P. Alliez, G. Ucelli, C. Gotsman, and M. Attene, "Recent advances in remeshing of surfaces," in *Shape Analysis and Structuring*, New York, NY, USA: Springer, 2008, pp. 53–82.
[10] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, 2nd ed. Hoboken, NJ, USA: Wiley, 2000.
[11] S.-W. Cheng, T. K. Dey, and J. R. Shewchuk, *Delaunay Mesh Generation*, Boca Raton, FL, USA: CRC Press, 2012.
[12] G. Peyré and L. D. Cohen, "Geodesic remeshing using front propagation," *Int. J. Comput. Vis.*, vol. 69, pp. 145–156, 2006.
[13] H. Edelsbrunner and N. R. Shah, "Triangulating topological spaces," *Int. J. Comput. Geom. Appl.*, vol. 7, no. 4, pp. 365–378, 1997.
[14] Q. Du, M. D. Gunzburger, and L. Ju, "Constrained centroidal Voronoi tesselations for surfaces," *SIAM J. Sci. Comput.*, vol. 24, no. 5, pp. 1488–1506, 2003.
[15] P. Alliez, D. Cohen-Steiner, M. Yvinec, and M. Desbrun, "Variational tetrahedral meshing," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 617–625, 2005.
[16] S. Valette and J. M. Chassery, "Approximated centroidal Voronoi diagrams for uniform polygonal mesh coarsening," *Comput. Graph. Forum*, vol. 23, no. 3, pp. 381–389, 2004.
[17] V. Nivoliers, D.-M. Yan, and B. Lévy, "Fitting polynomial surfaces to triangular meshes with Voronoi squared distance minimization," in *Proc. 20th Int. Meshing Roundtable*, 2011, pp. 601–618.
[18] H. Pan, Y.-K. Choi, Y. Liu, W. Hu, Q. Du, K. Polthier, C. Zhang, and W. Wang, "Robust modeling of constant mean curvature surfaces," *ACM Trans. Graph.*, vol. 31, no. 4, p. 85, Jul. 2012.
[19] Z. Chen, Z. Yuan, Y.-K. Choi, L. Liu, and W. Wang, "Variational blue noise sampling," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 10, pp. 1784–1796, Oct. 2012.
[20] V. Nivoliers and B. Lévy, "Approximating functions on a mesh with restricted voronoi diagrams," *Comput. Graph. Forum*, vol. 32, no. 5, pp. 83–92, 2013.
[21] D.-M. Yan and P. Wonka, "Gap processing for adaptive maximal Poisson-disk sampling," *ACM Trans. Graph.*, vol. 32, pp. 148:1–148:15, 2013.
[22] D.-M. Yan and P. Wonka, "Adaptive maximal Poisson-disk sampling on surfaces," in *Proc. ACM SIGGRAPH Asia Tech. Briefs*, 2012, pp. 21:1–21:4.
[23] F. Sun, Y.-K. Choi, W. Wang, D.-M. Yan, Y. Liu, and B. Lévy, "Obtuse triangle suppression in anisotropic meshes," *Comput. Aided Geom. Des.*, vol. 28, no. 9, pp. 537–548, 2011.
[24] G. Leibon and D. Letscher, "Delaunay triangulations and Voronoi diagrams for riemannian manifolds," in *Proc. 16th Annu. Symp. Comput. Geom.*, 2000, pp. 341–349.
[25] J.-D. Boissonnat, R. Dyer, and A. Ghosh, "Constructing intrinsic Delaunay triangulations of submanifolds," Inria, Rocquencourt, France, Res. Rep. RR-8273, 2013.
[26] N. Amenta, M. Bern, and M. Kamvysselis, "A new Voronoi-based surface reconstruction algorithm," in *Proc. ACM 25th Annu. Conf. Comput. Graph. Interactive Techn.*, 1998, pp. 415–421.
[27] T. K. Dey, J. A. Levine, and A. Slatton, "Localized delaunay refinement for sampling and meshing," *Comput. Graph. Forum*, vol. 29, no. 5, pp. 1723–1732, 2010.
[28] D. Cohen-Steiner, P. Alliez, and M. Desbrun, "Variational shape approximation," *ACM Trans. Graph.*, vol. 23, pp. 905–914, 2004.
[29] A. Meyer and S. Pion, "FPG: A code generator for fast and certified geometric predicates," in *Proc. Real Numbers Comput.*, 2008, pp. 47–60.
[30] CGAL, Computational Geometry Algorithms Library, (2013), [Online]. Available: http://www.cgal.org
[31] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982.
[32] Y. Liu, W. Wang, B. Lévy, F. Sun, D.-M. Yan, L. Lu, and C. Yang, "On centroidal Voronoi tessellation — energy smoothness and fast computation," *ACM Trans. Graph.*, vol. 28, no. 4, pp. 101:1–101:17, 2009.
[33] A. Lagae and P. Dutré, "A comparison of methods for generating Poisson disk distributions," *Comput. Graph. Forum*, vol. 27, no. 1, pp. 114–129, 2008.
[34] P. Frey and H. Borouchaki, "Surface mesh evaluation," in *Proc. 6th Int. Meshing Roundtable*, 1997, pp. 363–374.
[35] G. Rong, M. Jin, L. Shuai, and X. Guo, "Centroidal voronoi tessellation in universal covering space of manifold surfaces," *Comp. Aided Geom. Des.*, vol. 28, no. 8, pp. 475–496, 2011.



Dong-Ming Yan received the bachelor's and master's degrees from Tsinghua University in 2002 and 2005, respectively, and the PhD degree from Hong Kong University in 2010. He is currently a research scientist at the King Abdullah University of Science and Technology (KAUST), and is an associate professor at the National Laboratory of Pattern Recognition of the Institute of Automation, Chinese Academy of Sciences (CAS). His research interests include computer graphics, geometric processing and visualization.



Guanbo Bao received the PhD degree in computer science from the Institute of Automation, Chinese Academy of Sciences, in 2012. He is currently an assistant professor at the Institute of Automation, Chinese Academy of Sciences. His research interests include high performance rendering, geometry processing and image editing.

Xiaopeng Zhang received the PhD degree in computer science from the Institute of Software, Chinese Academy of Sciences (CAS) in 1999. He is currently a professor at the National Laboratory of Pattern Recognition of the Institute of Automation, CAS. He received the National Scientific and Technological Progress Prize (second class) in 2004. His main research interests include computer graphics and image processing.

Peter Wonka is currently an associate professor at the King Abdullah University of Science and Technology (KAUST) and Arizona State University (ASU). His research interests include topics in computer graphics, visualization, computer vision, remote sensing, image processing, and machine learning.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.