A flexible framework for surface reconstruction from large point sets

Rémi Allègre *, Raphaëlle Chaine, Samir Akkouche

LIRIS UMR 5205 CNRS / Université Lyon 1, Villeurbanne, F-69622, France

Abstract

This paper presents a flexible method to reconstruct simplified mesh surfaces from large unstructured point sets, extending recent work on dynamic surface reconstruction. The method consists of two core components: an efficient selective reconstruction algorithm, based on geometric convection, that simplifies the input point set while reconstructing a surface, and a local update algorithm that dynamically refines or coarsens the reconstructed surface according to specific local sampling constraints.

A new data structure is introduced that significantly accelerates the original selective reconstruction algorithm and makes it possible to handle point set models with millions of sample points. This data structure mixes a kd-tree with the Delaunay triangulation of the selected points enriched with a sparse subset of landmark sample points. This design efficiently responds to the specific spatial location issues of the geometric convection algorithm. It also permits the development of an out-of-core implementation of the method, so that simplified mesh surfaces can be seamlessly reconstructed and interactively updated from point sets that do not fit into main memory.

Keywords: Surface reconstruction, geometric convection, point set simplification, dynamic level of detail update, out-of-core reconstruction.

1 Introduction

The recent advances in 3D scanning technologies have led to an increasing need for techniques capable of processing massive discrete geometric data. We consider the problem of modeling surfaces from unstructured point sets obtained from 3D acquisition devices such as range scanners. This task needs flexibility for three main reasons. First, the complexity of a model may have to change depending on application requirements and hardware limitations. This typically involves oversampling reduction, geometry simplification, and level of detail management. Further, a model should be easy to update when new data from additional scans or procedural scan completion becomes available, or when applying smoothing or denoising methods. Finally, one should be able to tune the parameter values involved in the algorithms to get the desired result without restarting the modeling process from scratch.

Most existing combinatorial surface reconstruction techniques produce static results that cannot be updated easily [1]. To go beyond this type of limitation, we have developed a dynamic framework with point set simplification and reconstruction local update abilities introduced in Allègre et al. [2] and based on a geometric convection algorithm. In this paper, we review the ideas and results presented in [3] that extend this work so that large data sets can be handled efficiently (Fig. 1). We present a data structure that significantly accelerates the original selective reconstruction algorithm to process point set models with millions of sample points. This

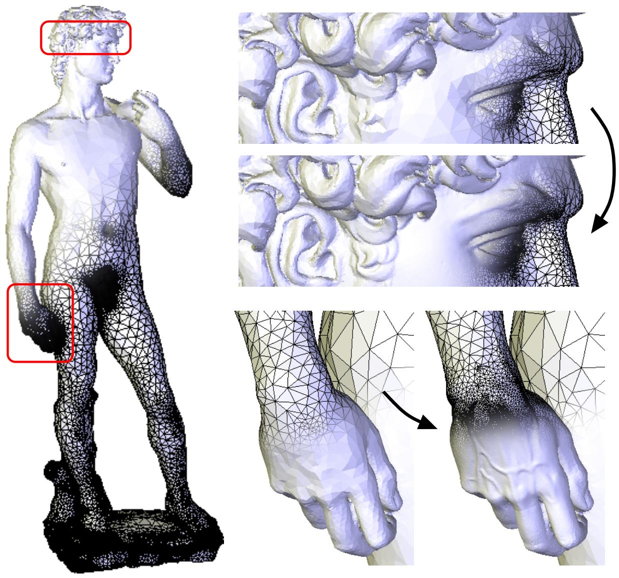

Figure 1: Dynamic surface reconstruction from a large point set model: DAVID (3.6M points). A simplified mesh was first reconstructed (left, 137k points, 4 minutes on Pentium IV 3GHz). Then the result was locally refined on the right temple and hand (right, 175k points, 28 seconds).

*Corresponding author. Fax: +33 4 72 43 15 36.
E-mail address: `remi.allegre@liris.cnrs.fr`.
URL: `http://liris.cnrs.fr/remi.allegre`.

data structure mixes a kd-tree with the Delaunay triangulation of the selected points enriched with a subset of landmark sample points obtained from the kd-tree. This design efficiently responds to the specific spatial location issues of the geometric convection algorithm, and is much less expensive than maintaining a global Delaunay triangulation. We also introduce sampling considerations to guarantee the behavior of the simplification operations. We further explain how the framework presented can be extended to an out-of-core implementation of the method to reconstruct simplified mesh surfaces from point sets that do not fit into main memory. Our method involves neither stitching nor consistent orientation issues, but only the update abilities offered by the geometric convection algorithm. We demonstrate the effectiveness of our framework on various detailed scanned statues with several million sample points. Our method can reconstruct high-quality simplified triangulated surfaces in a few minutes. Geometric detail can then be recovered or reduced locally whenever needed in a few seconds, which can make the method useful for viewpoint-dependent surface reconstruction. The abilities offered by the framework presented undeniably contribute flexibility to the 3D shape reconstruction process with reasonable performance.

The remainder of this paper is organized as follows. We start with an overview of related research (Section 2). Then we present a detailed analysis of the original framework in order to highlight the key elements that deserve special care (Section 3). We also deal with the sampling properties that should be reflected by the point set to guarantee the local thickness measure used by the simplification procedure. Then we describe our accelerated selective reconstruction algorithm (Section 4) before generalizing to an out-of-core extension (Section 5).

2 Related work

In the last few years, a great deal of work has been carried out on surface reconstruction from data sets with millions of sample points, including unorganized point sets [4, 5, 6, 7] and sets of range images [8, 9]. These methods are often used to produce a triangulated mesh surface, which is a standard representation for fast visualization and geometry-processing algorithms. However, the data used to generate these meshes are generally overly dense, due to uniform grid sampling patterns, and a mesh simplification step is required for use in common applications.

Point set simplification techniques offer an alternative to the standard pipeline by introducing a simplification step before the reconstruction process. These techniques aim at reducing the redundancy of the input data in order to accelerate subsequent reconstruction or visualization. Subsampling algorithms decimate the point set [10, 11, 12] while resampling algorithms compute new point locations [13, 14, 15]. These techniques rely either on oriented normals and local connectivity information obtained from k-neighborhoods or on a global Delaunay triangulation or Voronoï diagram, which represents a significant part of a surface reconstruction process that would take all the points into account.

Several algorithms that perform reconstruction and simplification in a single framework have been studied recently. Boissonnat and Cazals [16] proposed a Delaunay-based coarse-to-fine reconstruction algorithm controlled by a signed distance function to an implicit surface. Ohtake et al. [7] developed an algorithm that resamples a point set using a quadric error metric, coupled with a specific fast local triangulation procedure. In both cases, the resulting sampling remains static, and their results were not extended to handle update operations, especially the removal of sample points when the level of detail needs to be lowered, or if additional data become available (e.g., when streaming data on a network, or during a digital acquisition project).

In Allègre et al. [2], we have tackled this limitation by devising a *dynamic surface reconstruction* framework in which the reconstruction becomes *selective* and *evolutive*. The originality of the approach is to integrate surface reconstruction, data simplification and dynamic data insertion or removal, e.g., for updating the level of detail, into a single framework. Starting from a dense unorganized input point set, we reconstruct a simplified triangulated surface by means of a Delaunay-based surface reconstruction algorithm called *geometric convection* [17] coupled with a local point set subsampling procedure. The Delaunay triangulation is constructed only for the sample points retained in order to maintain a history of the reconstruction process. The reconstructed surface can then be easily updated by inserting or removing sample points without restarting the reconstruction process from scratch. However, the method lacks an efficient data structure to handle large data sets.

3 Reconstruction framework analysis

In this section, we briefly review the classic geometric convection algorithm described in [17] and how it is embedded into a dynamic framework with simplification and update abilities, as developed in [2]. We focus on the geometric predicates and queries involved in the surface reconstruction algorithm.

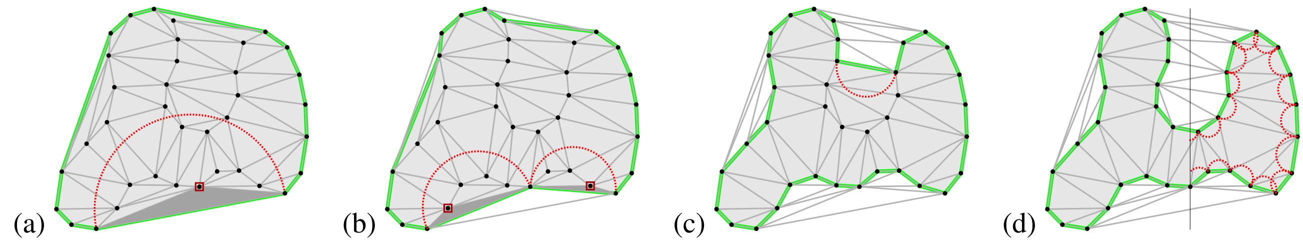

Figure 2: Geometric convection towards a 2D point set. In (a), an enclosing curve is initialized on the convex hull of the point set. The current edge, enclosed by a non-empty Gabriel half-ball, forms a Delaunay triangle (dark gray) with the square point. This triangle becomes external, the curve is updated (b), and it continues to shrink. In (c), an edge is found to block a pocket; it will be forced. The final result is shown in (d) with empty Gabriel half-balls.

3.1 Geometric convection

The geometric convection algorithm is a surface reconstruction algorithm that proceeds by filtering the Delaunay triangulation of an input point set sampled from a smooth surface [1]. This method has some similarities with the Wrap [18] and Flow Complex [19] techniques. The filtration is guided by a convection scheme related to level set methods [20] that consists in shrinking an enclosing surface under the influence of the gradient field of a distance function to the closest sample point. This process results in a closed, oriented triangulated surface embedded in the Delaunay triangulation of the point set, characterized by an *oriented Gabriel property* [17]. This means that for every facet, the diametral half-ball located inside the surface, or *Gabriel half-ball*, contains no sample point.

Let $P \subset \mathbb{R}^3$ denote the input point set and $\hat{S}$ the surface in convection. The convection scheme can be completely achieved through the Delaunay triangulation of P by removing the facets that do not meet the oriented Gabriel property through an iterative sculpting process that starts from the convex hull. The $\hat{S}$ surface is a closed triangulated surface that is maintained at every step, all the facets oriented *inwards*, and two meeting facets can collapse locally, which may change its topology. A local study (or a more global solution) is required to dig into *pockets* that may locally block the convection scheme, e.g., based on local granularity. The algorithm is illustrated on a 2D point set in Figure 2.

The geometric evolution of $\hat{S}$ through the convection process is guided locally by a geometric predicate P_{og} and a geometric query Q_{dt}, defined as follows:

(P_{og}) Given an oriented Delaunay facet **pqr**, test whether it satisfies the oriented Gabriel property.
(Q_{dt}) Given an oriented Delaunay facet **pqr**, find the point $\mathbf{s} \in P$ such that **pqrs** forms a Delaunay tetrahedron enclosed in the half-space above the facet.

The half-space above an oriented facet is the half-space facing the interior of the surface.

Assuming that the Delaunay triangulation of the input point set has been constructed, P_{og} and Q_{dt} are both evaluated in constant time, and the overall complexity of the algorithm is linear in the number of Delaunay cells traversed by the surface.

3.2 Selective reconstruction

Considering as a limitation that the original geometric convection algorithm takes all the data into account, the idea of our selective reconstruction is to associate it with a local subsampling procedure. In presence of an overly dense input point set, our goal is to produce a simplified triangulated surface that remains close to the sampled surface, up to an error tolerance, by adapting the sampling density to the local geometry variations.

Every time a new sample point $\mathbf{p} \in P$ is incorporated into the surface in convection $\hat{S}$, the idea is to remove the sample points in P that do not belong to $\hat{S}$ in a circular neighborhood around $\mathbf{p}$ whose radius reflects the local geometry. The geometric information held by $\mathbf{p}$ has to be sufficiently representative of the point positions in this neighborhood. The selective reconstruction algorithm therefore relies on a procedure that, given a sample point, detects and removes all the points in its neighborhood that are not geometrically significant. Redundancy is characterized by a radius that adapts to local curvature and thickness, and decimation is performed in a fraction of this radius, called the *simplification radius*. In practice, we compute the simplification radius based on a *local thickness measure*.

The remainder of this section is organized as follows. In Section 3.2.1, we formalize the notion of the local thickness measure that we introduced in [2, 21] to control the simplification radius. The sampling conditions under which the discrete estimation of this measure responds to certification are also examined. In Section 3.2.2, we then explain how the simplification radius is controlled, based on the local thickness measure. Section 3.2.3 provides a detailed analysis of the selective reconstruction algorithm in terms of geometric predicates and queries.

3.2.1 Local thickness measure

The simplification radius is computed using a local thickness measure that we introduced in [2, 21]. The goal is to decimate the point set according to the local geometry while maintaining good sampling conditions for the reconstruction algorithm. This requires a measure reflecting both the local curvature of the surface and the local thickness of the solid that it bounds. The distance to the medial axis takes these two aspects into account [22]. However, this distance is not easy to estimate from a point set [23, 24]. As an alternative, we developed a local thickness measure with a local definition and easy evaluation from a point set.

Continuous setting

Let S denote a closed surface. We wish to estimate the local thickness of the solid bounded by S at any point $\mathbf{x} \in S$. The main idea of our measure is to grow a ball $\mathcal{B}$ around $\mathbf{x}$ until a criterion related to curvature is met by $\mathcal{B} \cap S$, or until $\mathcal{B}$ touches S, which implies that $\mathcal{B} \cap S$ is no longer a topological disk. The radius of this ball is called *local thickness* at point $\mathbf{x}$. This principle is formalized by the following definition.

Definition 1 (Local thickness). *Given a closed surface S, the local thickness* $\mathrm{lt}(\mathbf{x})$ *at a point $\boldsymbol{x} \in S$ for a fixed geometric precision ρ_{geom} of the solid bounded by S is defined as:*

$$\mathrm{lt} : S \to \mathbb{R}_+,\ \mathbf{x} \mapsto \mathrm{lt}(\mathbf{x}) = \min\{r_{geom}(\mathbf{x}), r_{topo}(\mathbf{x})\}$$

where $r_{geom}(\mathbf{x})$ and $r_{topo}(\mathbf{x})$, respectively, denote a geometric radius and a topological radius at point $\mathbf{x}$. Let $\mathrm{n}(\mathbf{x})$ denote a unit normal vector at a point $\mathbf{x} \in P$ (regardless of its orientation). These two radii are defined as follows:

- *$r_{geom}(\mathbf{x})$ is the distance from $\mathbf{x}$ to the closest point $\mathbf{y} \in S$ such that:*

$$|\mathrm{n}(\mathbf{y}) \cdot \mathrm{n}(\mathbf{x})| \leq \rho_{geom},\ \rho_{geom} \in [0,1]$$

- *$r_{topo}(\mathbf{x})$ is the distance from $\mathbf{x}$ to the closest point $\mathbf{y} \in S$ such that the ball $\mathcal{B}(\mathbf{x}, \|\mathbf{xy}\|) \cap S$ is not a topological disk.*

The closeness between two points on the surface S is considered in the sense of the Euclidean distance. The geometric radius r_{geom} reflects the thickness in regions where the surface is highly curved. Curvatures are not explicitly estimated, but rather we are studying the variations of normal directions to measure the local curvature of the surface, which will avoid some complex calculations in the discrete setting. The geometric radius is thus computed as the distance to the closest point such that the deviation of the normal direction exceeds a certain threshold. The ρ_{geom} value is the level of contrast that we wish to preserve when estimating the thickness, which determines the tolerated variation range of the normal directions in the neighborhood of the measure point with respect to the normal direction at this point. This makes it possible to take into account the smallest features of the surface, or, in contrast, to consider the geometry variations at a more global scale, which may be important in practice to process noisy data.

The r_{topo} radius bounds the local thickness measure to a topological disk on the surface. This radius reflects the distance between the surface patch of the measure point and the closest one. Assuming that S is a smooth surface, the radius $r_{topo}(\mathbf{x})$ at a point $\mathbf{x}$ can also be defined as the distance from $\mathbf{x}$ to the closest point $\mathbf{y} \in S$ such that the sphere centered at $\mathbf{x}$ with radius $r_{topo}(\mathbf{x})$ is tangent to S at $\mathbf{y}$, i.e., $|\mathrm{n}(\mathbf{y}) \cdot \frac{\mathbf{xy}}{\|\mathbf{xy}\|}| = \rho_{topo}$ with $\rho_{topo} = 1$. Thus r_{topo} can be evaluated analytically, and to tolerance it by tuning the ρ_{topo} to values lower than 1. This also makes it possible to devise an algorithm considering normal directions in the discrete setting.

Figure 3 illustrates the local thickness at two points on a smooth planar curve. At $\mathbf{x}_1$, the minimum radius is the topological one, whereas at $\mathbf{x}_2$, the minimum radius is the geometric one. It should be noted that the geometric and topological radii are not necessarily relevant when considered independently. For instance, depending on the local curvature, the value of the geometric radius can be influenced by the proximity of another surface patch. In this case, the topological radius takes over from the geometric radius. Conversely, the criterion controlling the topological radius may be satisfied by a point that is too far away from the measure point,

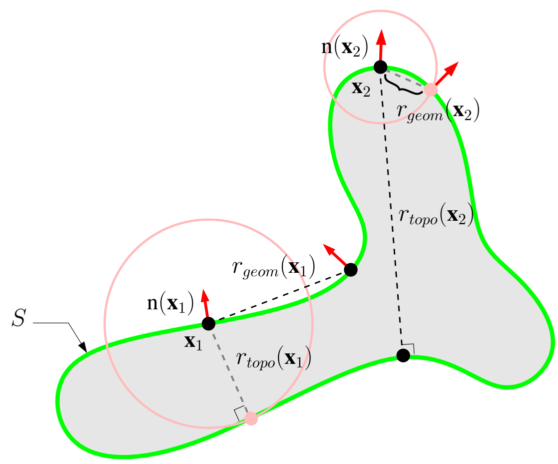

Figure 3: Local thickness measured on a smooth planar curve. The radii of the two circles represent the local thickness lt at two points $\mathbf{x}_1$ and $\mathbf{x}_2$.

so that the geometric radius is the most relevant.

The local thickness can be computed analytically for surfaces or curves in an explicit or parametric representation (Fig. 4, top). Because of its local nature, this measure is not easy to link to the local feature size as defined by Amenta and Bern [22], although they may be close to each other in some cases (Fig. 4, bottom). A different but related idea has been developed by Boissonnat and Oudot [25] in a Lipschitz surface sampling framework. A direct link with our study is the definition of a radius at any point that takes the local variations of the surface into account. Given a surface S, they define the k-Lipschitz radius at a point $\mathbf{p}$ as the radius of the largest ball $\mathcal{B}$ centered at $\mathbf{p}$ such that $S \cap \mathcal{B}$ is the graph of a k-Lipschitz bivariate function. They prove that guarantees similar to those obtained with ε-samples of smooth surfaces can be obtained from a sample of a Lipschitz surface S such that any point of S has a sample point at a distance less than a fraction of the Lipschitz radius of S. However, the authors do not discuss the implementation of their measure. In our study, we are more concerned with the estimation of our measure in the discrete setting so as to propose a simple algorithm applicable to a point sample on an unknown surface.

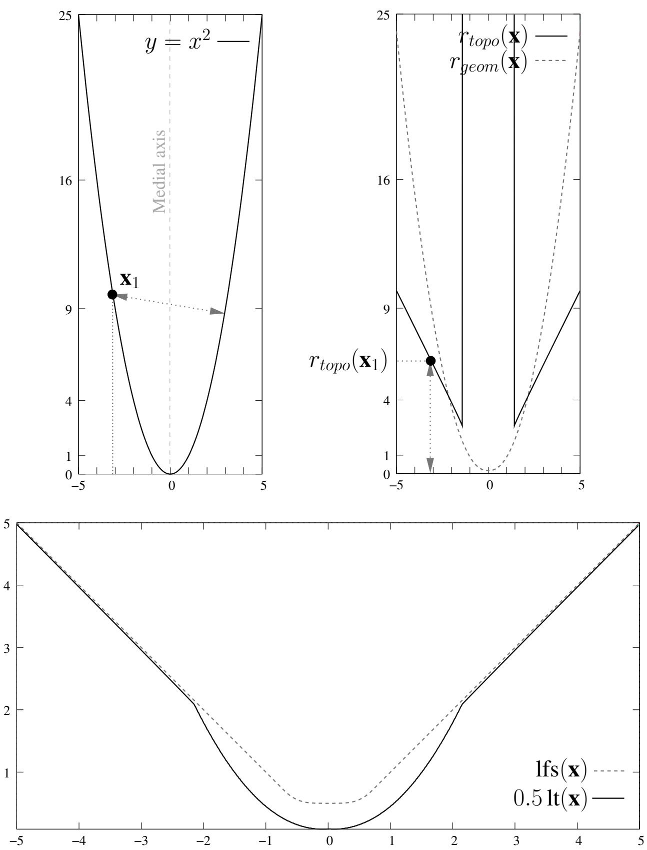

Figure 4: Local thickness for $\rho_{geom} = 0.95$ and local feature size measured on a parabola with equation $y = x^2$. For this open curve, $r_{topo}(\mathbf{x})$ with $\mathbf{x} = (x, x^2)$ is considered as infinite for $x \in [-\sqrt{2}, \sqrt{2}]$. In this interval, the local thickness is the geometric radius.

Discrete setting

This local thickness measure can be directly extended to a finite point sample on a closed smooth surface S equipped with surface normal directions.

Definition 2 (Discrete local thickness). *Given a point sample P on a closed smooth surface S, the local thickness $\tilde{\mathrm{lt}}$ at a point $\mathbf{p} \in P$ for fixed precisions ρ_{geom} and ρ_{topo} is defined as:*

$$\tilde{\mathrm{lt}} : P \rightarrow \mathbb{R}_+,\ \mathbf{p} \mapsto \tilde{\mathrm{lt}}(\mathbf{p}) = \min\{\tilde{r}_{geom}(\mathbf{p}), \tilde{r}_{topo}(\mathbf{p})\}$$

where the radii $\tilde{r}_{geom}(\mathbf{p})$ and $\tilde{r}_{topo}(\mathbf{p})$ at point $\mathbf{p}$ are defined as follows:

- *$\tilde{r}_{geom}(\mathbf{p})$ is the distance from $\mathbf{p}$ to the closest point $\mathbf{q} \in P$ such that:*
$$|\mathrm{n}(\mathbf{q}) \cdot \mathrm{n}(\mathbf{p})| \leq \rho_{geom},\ \rho_{geom} \in [0,1]$$
- *$\tilde{r}_{topo}(\mathbf{p})$ is the distance from $\mathbf{p}$ to the closest point $\mathbf{q} \in P$ such that:*
$$|\mathrm{n}(\mathbf{q}) \cdot \frac{\mathbf{pq}}{\|\mathbf{pq}\|}| \geq \rho_{topo},\ \rho_{topo} \in [0,1]$$

The definition is similar to the one in the continuous setting, the discrete topological radius $\tilde{r}_{topo}$ being toleranced by the ρ_{topo} parameter.

The validity of this definition, and its relation with the continuous setting, depends on the properties of P with respect to S. In the next section, we examine the sampling conditions under which the discrete local thickness measure is well defined.

Sampling conditions

We determined sampling conditions guaranteeing that the discrete local thickness is defined and consistent with the continuous measure. We provide the conditions independently for the geometric radius (Condition 1) and for the topological radius (Condition 2). See Figures 5 and 6 for illustrations.

Condition 1 *There exists a constant $C \geq 1$ such that for every point $\mathbf{x} \in S$, if there exists a point $\mathbf{y} \in S$ such that $\arccos(|\mathrm{n}(\mathbf{x}) \cdot \mathrm{n}(\mathbf{y})|) > C \arccos(\rho_{geom})$, then there exists a point $\mathbf{p} \in P$ such that $\|\mathbf{xp}\| \leq \|\mathbf{xy}\|$ and $|\mathrm{n}(\mathbf{x}) \cdot \mathrm{n}(\mathbf{p})| \leq \rho_{geom}$.*

If Condition 1 is satisfied by the point sample P, this means that for any point $\mathbf{x} \in S$, the discrete radius $\tilde{r}_{geom}(\mathbf{x})$ toleranced by ρ_{geom} is smaller than the continuous radius $r_{geom}(\mathbf{x})$ toleranced by $\cos(C \arccos(\rho_{geom}))$.

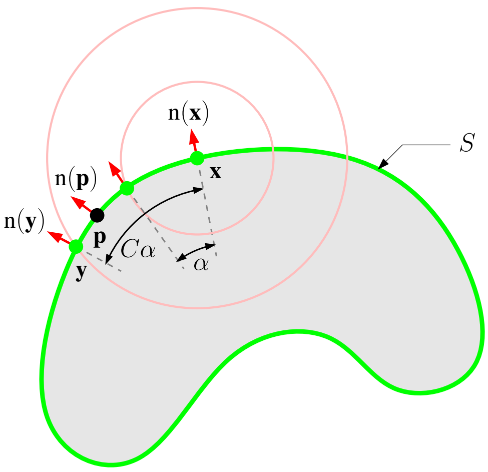

Figure 5: Illustration of Condition 1. On the figure, α represents the angle $\arccos(\rho_{geom})$.

Condition 2 *For every point* $\mathbf{x} \in S$*, if there exists a point* $\mathbf{y}$ *such that* $\mathcal{B}(\mathbf{x}, \|\mathbf{xy}\|) \cap S$ *is not a topological disk, let* $\mathbf{z} \in S$ *be the closest point from* $\mathbf{y}$ *such that* $|\mathrm{n}(\mathbf{z}) \cdot \frac{\mathbf{xz}}{\|\mathbf{xz}\|}| < \rho_{topo}$*. Then there exists a point* $\mathbf{p} \in P$ *such that* $\|\mathbf{xp}\| \leq \|\mathbf{xz}\|$ *and* $\mathcal{B}(\mathbf{x}, \|\mathbf{xp}\|) \cap S$ *is not a topological disk.*

This condition means that the radius $\tilde{r}_{topo}$ parameterized by ρ_{topo} is lower than r_{topo} parameterized by ρ_{topo}.

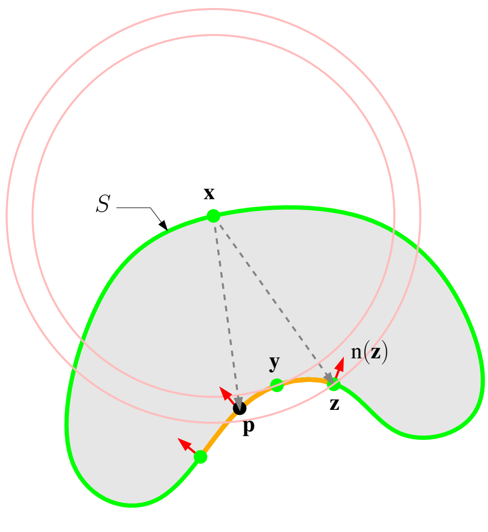

Figure 6: Illustration of Condition 2.

These sampling conditions imply a minimum density of the point sample that is linked to the parameters ρ_{geom} and ρ_{topo}. However, it is worth mentioning that there is no upper bound, i.e., the sampling can be denser than necessary without altering the local thickness estimation.

In practice, one important question is how to choose the values for the ρ_{geom} and ρ_{topo} parameters. In our selective reconstruction framework, the value of ρ_{geom} is user-specified. The value of ρ_{topo} should be set according to the sampling conditions. However, given a point set, it is hard to check whether it satisfies these conditions. In all our tests on merged range scans, which are generally very dense, we used $\rho_{topo} = 0.9$. Ways to compute ρ_{topo} for a given point set, as well as the relationship between the geometric and topological radii warrant further investigation and will be addressed in future work.

3.2.2 Simplification radius control

Based on the previous discrete local thickness measure, the simplification radius for a sample point $\mathbf{p} \in P$ is computed as a fraction of the local thickness at this point:

$$r_{simp}(\mathbf{p}) = \alpha.\tilde{\mathrm{lt}}(\mathbf{p}), \alpha \in [0, 1]$$

where $\alpha \in [0, 1]$ is a factor that controls the density of the point sample that results from the selective reconstruction algorithm and its distribution near sharp features. We call this factor the *anticipation factor*. Its effect is illustrated in Figure 7. Decimating with $\alpha = 1$ results in skinny triangles near sharp features, where the sampling density increases too rapidly. Setting α to a value that is less than 1 makes it possible to anticipate these variations in the subsampling process and to obtain a smooth density gradient near sharp features, as shown in Figure 8.

This nearly results in a locally uniform sampling and triangles with a good aspect ratio. In practice, we set its value to 0.5, which generally provides a good tradeoff between sampling density and triangle quality. After the simplification, the sampling distribution will only reflect the local thickness of the object, which is intuitively related to the local feature size. This sampling is consequently more predisposed to surface reconstruction algorithms than an initial overly dense sampling.

3.2.3 Algorithm analysis: predicates and queries

In addition to the previously defined P_{og} predicate and Q_{dt} query, the selective reconstruction algorithm requires a query Q_{nn} defined as follows:

(Q_{nn}) Given a point $\mathbf{p} \in P$, incrementally return its nearest neighbors in P.

If $\hat{R} \subset P$ denotes the set of removed sample points at a given time, then P_{og}, Q_{dt}, are evaluated within $P \backslash \hat{R}$ at that time. Since many sample points may be discarded, constructing the Delaunay triangulation of the entire input point set may be uselessly expensive. In this case, other predicates and queries are needed to make the triangulated surface evolve on the fly. Evaluating P_{og} now involves an additional query Q_{hb} and a predicate P_{ct} defined as follows:

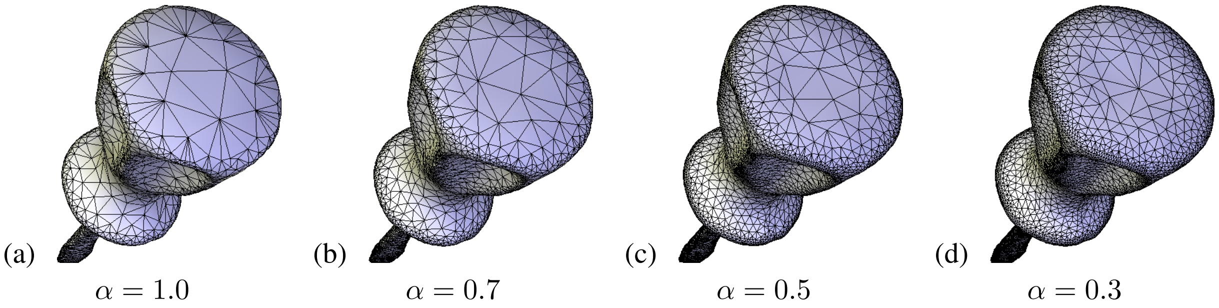

Figure 7: Selective reconstruction results obtained with decreasing anticipation factors (fixed ρ_{geom} and ρ_{topo}).

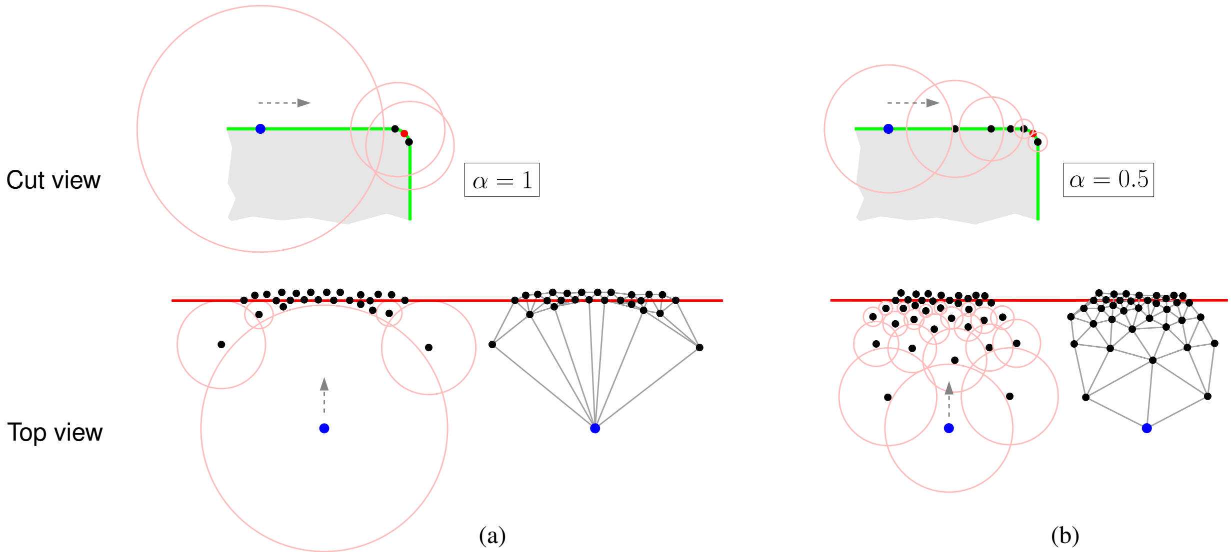

Figure 8: Effect of the anticipation factor on the sampling distribution near a sharp edge. Sample points are represented with their simplification regions. In (a), $\alpha = 1$; the sampling density after the selective reconstruction process changes rapidly near the edge, which results in skinny triangles and high valence vertices. In (b), $\alpha = 0.5$; the sampling density changes more progressively. The arrows show the direction of the shrinking surface.

($\mathsf{Q_{hb}}$) Given a facet **pqr** $\in \hat{S}$, report the points in $P \setminus \hat{R}$ located inside the Gabriel half-ball of **pqr**.
($\mathsf{P_{ct}}$) Given a point **p** $\in P \setminus \hat{R}$, test whether it conflicts with a tetrahedron.

The question of how to efficiently evaluate $\mathsf{P_{og}}$, $\mathsf{Q_{dt}}$, and $\mathsf{Q_{nn}}$ arises. A kd-tree data structure was used in [2] to report the sample points located inside the Gabriel half-ball of a facet and to search the nearest neighbors of a sample point. For a facet that does not satisfy $\mathsf{P_{og}}$, the search space for $\mathsf{Q_{dt}}$ can be reduced to its Gabriel half-ball. However, these half-balls may contain a large part of the input point set, especially at the beginning of the reconstruction process (see Figure 2(a), for example). Moreover, when $\mathsf{P_{og}}$ is satisfied and a pocket is detected, the search space for $\mathsf{Q_{dt}}$ can extend to the entire half-space above the facet.

The main limitation of the algorithm regarding performance is the lack of visibility of "what lies ahead" of the evolving surface in the unexplored domain during the convection process. To handle large data sets efficiently, a better localization of geometric queries is required. In Section 4, we show that performance can be considerably improved by dynamically maintaining a partial Delaunay triangulation of the data.

3.3 Local update

The geometric convection technique (selective or not) can be used locally to update the reconstructed surface. This is useful to add or remove data, or to change the level of detail of the reconstruction. We proposed a local reconstruction update algorithm that uses the Delaunay triangulation of the sample points retained in the reconstructed surface. This triangulation maintains information on the relative order in which the surface traversed the Delaunay cells. When inserting or removing data, the reconstruction history is locally invalidated in a conflict region, and the locality of the convection process is then exploited to update the reconstruction and restore the history.

During the reconstruction process, constructing the

Delaunay triangulation of the *retained* sample points makes it possible to locally update the reconstructed surface by adding or removing sample points in a dynamic fashion. This functionality takes advantage of the *discovery relation* induced by the convection scheme on the set of Delaunay cells traversed by the surface. This relation is stored in the cells that have been visited (Fig. 9). When inserting sample points, these points conflict with a set of Delaunay cells that form a conflict region. This region is retriangulated and the discovery relation between Delaunay cells is restored by restarting the reconstruction process from its boundary parts located outside the current surface. The surface can step back locally when some cells can no longer be discovered. When removing sample points, the conflict region cells region are the cells attached to the points to be removed. More details are given in [2].

To locally change the level of detail of a reconstructed surface, a region of interest and a local ρ_{geom} value are first defined. All the sample points removed in this region are rehabilitated, and the Delaunay cells whose circumsphere intersects the region of interest form the conflict region. The internal Delaunay vertices are removed and the selective reconstruction process restarts as described above, by taking the local simplification parameter into account.

4 Efficient implementation

This section describes how the selective reconstruction can be made efficient and appropriate for large data sets. Our first goal is to accelerate the evaluation of the previously mentioned P_{og} predicate and Q_{dt} query. This is achieved by first structuring and reducing the search space covered by these operations, based on a partial Delaunay triangulation of the input point set. Spatial search is then carried out through a kd-tree data structure with an optimized algorithm. At the end of the section, we describe an out-of-core selective reconstruction algorithm that mixes the in-core technique with the local update algorithm to handle point sets that do not fit into memory.

4.1 Data structure and accelerated algorithm

An appropriate data structure is required to efficiently evaluate the oriented Gabriel predicate, the Delaunay cell query, and the nearest neighbor query involved in the selective reconstruction algorithm. We rejected the possibility of using a global Delaunay triangulation of the input point set, and we underlined that the use of a kd-tree only is not a perfect alternative. It is well suited for nearest neighbor queries. However, it can be heavy to use for oriented Gabriel predicate evaluations, and it is clearly not adapted for Delaunay cell queries.

Without some structure information on the domain bounded by the surface in convection, all the points contained in the Gabriel half-balls must be reported by the kd-tree and tested to respond to the Delaunay cell queries. The search domains can be very large, especially at the beginning of the reconstruction process, which is inefficient. To reduce these search domains, our solution combines a kd-tree with the Delaunay triangulation of the sample points retained in the evolving surface, since it is already used for update purposes. We enrich the unexplored part of this triangulation with landmark points that help structure the unexplored domain.

Data structure Let us consider the Delaunay triangulation of the retained sample points at one step of the reconstruction algorithm. Every facet of the shrinking surface is the interface between two cells; we call *front cell* the one that is enclosed in the surface. Front cells are connected to opposite vertices on the surface and give information on the extent of the unexplored domain. However, the part of their circumsphere located inside the surface can enclose a larger spatial domain than the Gabriel half-balls. The Delaunay triangulation of the retained sample points is therefore not sufficient to reduce the search space for the spatial queries involved in the convection algorithm: additional sample points are required to "break" large front cells.

We begin with a set of landmark sample points obtained from a kd-tree structure with a threshold on the maximum number of points per leaf. In every leaf cell, the point that is the closest to the centroid is retained as a landmark (their density will be discussed later). The Delaunay triangulation $\hat{D}$ of these points is then built and enriched with the corners of a bounding box. The surface is initialized on the the bounding box, and the reconstruction process can then be run benefiting from *smaller* front cells that will help to accelerate the evaluation of both P_{og} and Q_{dt}. In parallel, spatial search is delegated to a kd-tree data structure that stores the entire input point set, with a specific algorithm that will be described later.

Accelerated algorithm The accelerated algorithm dynamically updates the Delaunay triangulation $\hat{D}$ throughout the reconstruction process by inserting retained sample points and removing unretained landmarks so that the latter do not affect the final result. Here we exploit the property that the Delaunay cells that become external to the surface remain until the end of the process, which is not the case for internal cells. Ex-

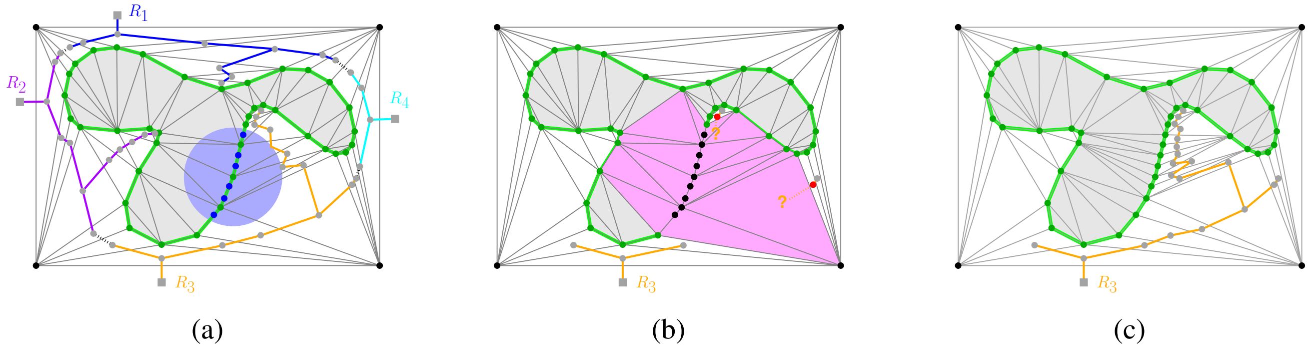

Figure 9: Local update on a curve reconstructed by geometric convection from a 2D point set. The result of the initial reconstruction is shown in (a) with the four discovery relations on the cells of the Delaunay triangulation that were traversed (R_i). A circular refinement region has also been defined. In (b), the sample points located in this region conflict with the set of colored cells. The R_3 relation is broken; the reconstruction process restarts locally. In (c), the reconstruction has been updated.

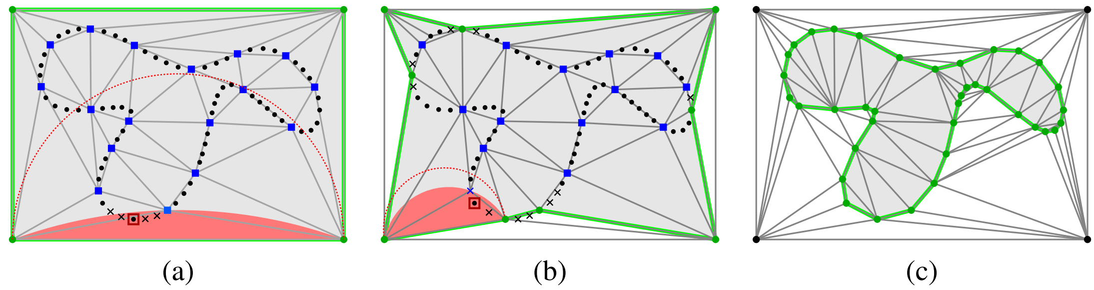

Figure 10: Accelerated selective reconstruction towards a 2D point set. In (a), an enclosing curve is initialized on a bounding box. A Delaunay triangulation has been built from its corners and a set of landmarks (filled square points). The current edge is enclosed by a non-empty Gabriel half-ball: the front vertex lies inside. Then, the point that forms a Delaunay triangle with the edge is searched within the disk that circumscribes the front cell (the square-dot point). The point set is then locally decimated around the point retained (cross points). In (b), the point retained was inserted in the triangulation and the curve continues to shrink. The facets attached to the corners of the bounding box are forced. The final result is shown in (c).

ternal cells are naturally protected from any subsequent vertex insertion or removal.

We continue with the notations of Section 3.1 to describe the algorithm. An illustration in 2D is provided in Figure 10. To check whether an oriented facet **pqr** of the surface $\hat{S}$ satisfies $\mathsf{P_{og}}$, we consider its front cell σ in the current Delaunay triangulation; its circumsphere is denoted as $\mathcal{S}$. We call *front vertex* the vertex of σ that is opposite the facet; its position is denoted as **s**. The Gabriel half-ball of the facet is finally denoted as $\mathcal{B}$, and the half-space above the plane that supports it as $\mathcal{H}$. The first step to evaluate $\mathsf{P_{og}}$ is to check whether **s** lies inside or outside $\mathcal{B}$.

1. If $\mathbf{s} \in \mathcal{B}$, then the $\mathsf{P_{og}}$ predicate is not satisfied. The $\mathsf{Q_{dt}}$ query is then performed in $(P \setminus \hat{R}) \cap \mathcal{S} \cap \mathcal{H}$, which corresponds to the set of points that conflict with σ. If this set is empty, then **pqrs** forms a Delaunay tetrahedron in $P \setminus \hat{R}$.

2. If $\mathbf{s} \notin \mathcal{B}$, it it not guaranteed that $\mathsf{P_{og}}$ is satisfied. To evaluate the predicate, we first obtain all the points in the set $(P \setminus \hat{R}) \cap \mathcal{B}$ through $\mathsf{Q_{hb}}$. If this set is not empty, then $\mathsf{P_{og}}$ is not satisfied and $\mathsf{Q_{dt}}$ is then performed in the set $(P \setminus \hat{R}) \cap \mathcal{B}$.

In the case where $\mathsf{P_{og}}$ is satisfied but a pocket is detected, then $\mathsf{Q_{dt}}$ is performed in the set of points $(P \setminus \hat{R}) \cap \mathcal{S} \cap \mathcal{H}$ that conflict with σ. If this set is empty, then **pqrs** forms a Delaunay tetrahedron in $P \setminus \hat{R}$.

Every time a new Delaunay tetrahedron is formed from a facet **pqr** and a point **x**, then **x** is inserted into the Delaunay triangulation provided $\mathbf{x} \neq \mathbf{s}$, and the surface is updated. Note that any facet attached to vertices of the bounding box should be opened, i.e., the query $\mathsf{Q_{dt}}$ should be performed, even when the predicate $\mathsf{P_{og}}$ is satisfied.

We now discuss the choice of the landmark points. The main benefit of these points is at the beginning of the process, where Gabriel half-balls may contain many sample points. As their size decreases, this benefit also diminishes, because the density of these points becomes insufficient. However, small Gabriel half-balls can be processed more efficiently. If this density is too high, then a great deal of time may be spent removing undesired landmarks. As the final simplification rate depends to a large extent on the shape and on the ρ_{geom} value, the optimal number of landmarks is not easy to determine. In practice, choosing one landmark for a few thousand points (between 1k and 10k) is sufficient to limit the spheres that circumscribe front cells to a few hundred points in the worst case and obtain a significant acceleration of the selective reconstruction process.

4.2 Accelerated spatial search

In the accelerated selective reconstruction algorithm, the $\mathbf{P_{og}}$ predicate is first evaluated by localizing subsets of sample points that conflict with front cells or that fall into Gabriel half-spheres. When the returned set of points is not empty for a given facet, then the point that forms a Delaunay tetrahedron with the facet has to be found ($\mathbf{Q_{dt}}$). Without information on the structure of the input point set, every point in this set is a potential candidate and thus needs to be tested. To reduce the number of tests, we order them based on a kd-tree data structure.

We first focus on the simple case where a facet of the surface is such that its front vertex is located inside the Gabriel half-ball of the facet. We start by searching for the non-empty leaves of the kd-tree that are likely to contain points that fall within the region bounded by the circumsphere of the front cell, restricted to the half-space defined by the facet; we call this region $\mathcal{C}$. This is achieved through a depth-first traversal of the kd-tree. If a kd-tree cell lies completely inside $\mathcal{C}$, then the leaves of the corresponding sub-tree are returned. The leaves that intersect $\mathcal{C}$ only partially are also returned. Testing whether a kd-tree cell intersects $\mathcal{C}$ involves two predicates: a sphere/box overlap test and half-space/box overlap test [26]. A counter that gives the number of remaining points in a leaf avoids testing empty kd-tree cells.

When non-empty leaves are reported, the next goal is to obtain the point that forms a Delaunay tetrahedron with the facet, with an average complexity better than linear in the number of points contained in the leaves. Our algorithm proceeds incrementally, starting with the sample point that maps to the front vertex of the facet as a candidate. The set of leaves reported for the facet are stored in a queue denoted as L, and the facet is denoted as **pqr**.

1. While L contains more than one element:

   (a) Take one point in each kd-tree leaf of L that falls into $\mathcal{C}$, if existing. Let M denote this set of points.

   (b) Search M for the best point candidate **c**, that is the point such that the circumsphere of **pqrc** contains no other point of M, based on $\mathbf{P_{ct}}$.

   (c) Remove from L the empty cells and the cells that do not conflict with tetrahedron **pqrc**.

2. Search for the best candidate from the remaining points.

The case where the facet has its opposite vertex outside its Gabriel half-ball is treated in a similar fashion, except that conflicts are first tested within the reported leaves that intersect the Gabriel half-ball in order to determine whether the facet satisfies $\mathbf{P_{og}}$. As soon as the predicate is found to be unsatisfied or if a pocket is detected, then the search is pursued in order to find the Delaunay candidate.

The method rapidly discards outlier leaves, i.e., those that are the least likely to contain the right candidate. However, it is often difficult to decide between the remaining leaves, since the candidates can ”jump” from one leaf cell to another. When the number of remaining leaves stagnates, we stop the process and switch to a linear search among the remaining points in order to prevent any computational overhead of testing conflicts between leaf cells and triangulation cells. In practice, the overall gain per facet is typically 10% to 20% of conflict tests between a point and a tetrahedron ($\mathbf{P_{ct}}$).

5 Out-of-core selective reconstruction

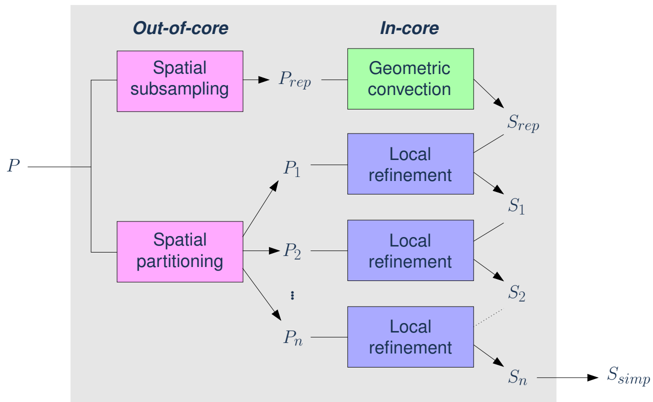

Figure 11: The pipeline of our out-of-core selective reconstruction algorithm.

Starting from a large and dense input point set that cannot be stored in main memory, our goal is to produce a simplified triangulated surface that fits into memory. A common strategy to simplify large unstructured meshes that cannot be entirely loaded into memory consists in partitioning the input data into

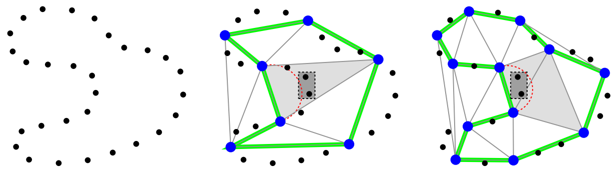

Figure 12: Accessible new data vs. inaccessible new data in 2D. We consider the input point set (a). In (b) and (c), the two new points in the rectangle are loaded. In (b), these points cannot be reached with the reconstruction (bold curve) from P_{rep} (bold points), whereas they can be reached in (c) where the reconstruction is finer.

clusters and then processing each one independently in-core [27, 28, 29]. This strategy does not extend easily to surface reconstruction from large unorganized point sets. Since no connectivity information between the different parts is available, stitching and orientation issues arise [5]. In our framework, we propose to circumvent this problem by maintaining some kind of global connectivity information based on the Delaunay triangulation of a subset of representative points, and process each cluster independently through the local update algorithm. Our algorithm proceeds in three steps that are schematized in Fig. 11 and summarized below:

1. The input point set P is filtered through a regular grid to obtain a subset of representative sample points P_{rep} and a partition of P into clusters $P_1 \cup P_2 \cup \ldots \cup P_n = P$.

2. The Delaunay triangulation of P_{rep} is built and the classic geometric convection algorithm is run on this point set.

3. For every subset P_i, the points that it contains are loaded into memory and then the reconstruction is locally refined in the corresponding region of space using the local update and selective reconstruction algorithms.

While partitioning the input point set in the first step, we wish to quickly extract a reduced set of representative points giving an approximate idea of the global shape. This sample is then used in the second step to produce a coarse reconstruction. This reconstruction step builds a discovery relation between a set of Delaunay cells that partition the entire data domain. This relation will be the basis for subsequent local reconstruction updates. Even if the initial reconstruction is not topologically correct and misses a few small features, this will not affect the quality of the final result; errors will be automatically fixed by local updates. However, from a computational point of view, it is preferable to start these updates with a sufficiently precise reconstruction. Indeed, major revisions of the surface may be expensive, both in time and memory. In order to limit them, the final surface should be *accessible* from the surface reconstructed from P_{rep} in the sense that it should be enclosed in the union of Gabriel half-balls of the shrinking surface (Fig. 12). Note that this condition is not mandatory to obtain a correct reconstruction. In practice, we simply filter the input point set on a grid with a fixed resolution.

The initial filtering and partition step is achieved by reading the input point set three times. During the first pass, we compute the smallest axis-aligned bounding box, which we next subdivide into a regular grid. In the second pass, for each non-empty grid cell the sample point that is the closest from the center is computed. This set of sample points forms the set P_{rep}. During this pass, we also count how many points fall into each grid cell. We next define a recursive binary partition of the grid structure with a user-specified maximum number of sample points per leaf; each leaf cell represents a cluster P_i. The maximum population threshold for each cluster should be set according to the amount of memory available on the target machine. During the third pass, the points are distributed among the different leaf cells. Depending on their number, the content of the clusters may be written in separate files on disk, or they may be filled and processed one at a time, which requires additional reading passes.

In Step 3, for each cluster P_i the set of cells of the current Delaunay triangulation that conflict with its points must be determined. To avoid multiple point locations in the Delaunay triangulation, conflicts are tested against the smallest axis-aligned bounding box of the points in P_i. We search for the Delaunay cells whose circumsphere intersects this bounding box. This is achieved by first locating the Delaunay cell that contains the center of the box and then extending the conflict region by recursively testing the neighboring cells. The result is a connected set of Delaunay cells that is used to initialize the local reconstruction update process. For spatial search queries, we construct a kd-tree from all the points inside the conflict region. This set includes P_i and may also include some points outside P_i attached to Delaunay cells in conflict with the bounding box of P_i, which guarantees that the different refined parts correctly merge together. The local update process is then achieved, as described in Section 3.3.

Two steps of the reconstruction process are illustrated on the LUCY model (14M points) in Figure 13. The boundaries of the different parts may be visible in the final result. However, the method produces no discontinuity in the sampling density. These boundaries can be completely eliminated by simply enlarging the clusters so that they contain neighboring sample points up to a distance that depends on the simplification parameter ρ_{geom} (Fig. 18).

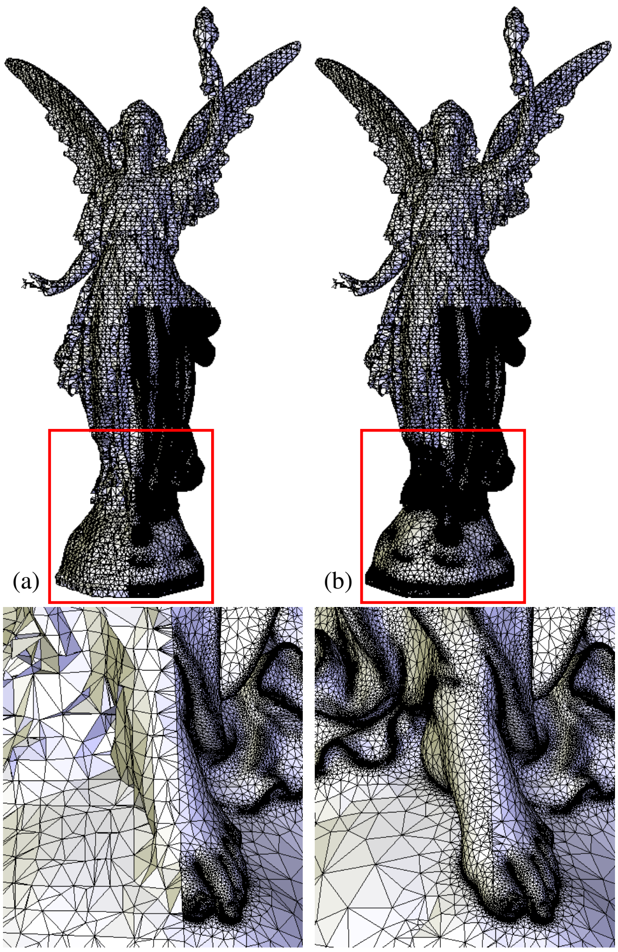

Figure 13: Two reconstruction steps for the LUCY model. In (a), the initial surface has been reconstructed from the representative points (25k) and a first local update step has been performed (bottom-left). In (b), one more cluster has been loaded and the reconstruction has been updated.

6 Results and performance

We implemented our extended dynamic surface reconstruction framework in C++ on a Linux platform using the Computational Geometry Algorithm Library, CGAL [30]. We use CGAL for constructing Delaunay triangulations and rely on filtered predicates for robust conflict tests.

Here we demonstrate the effectiveness of our framework on several large point-set models that were obtained from laser-range scanning

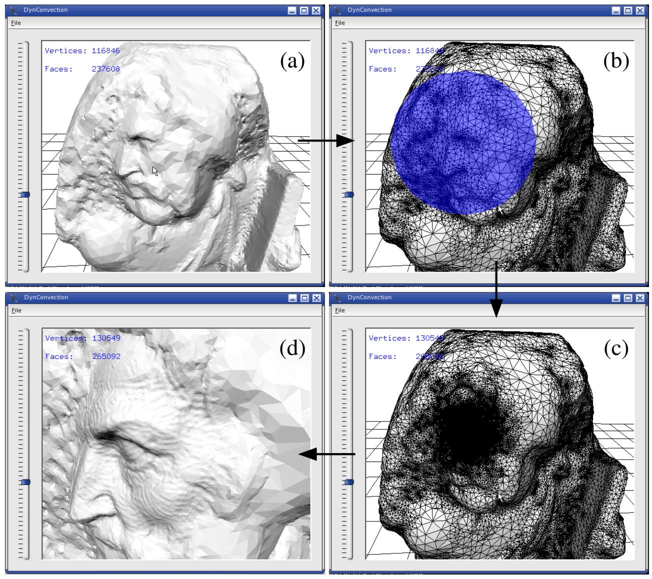

Figure 14: Some screenshots of our dynamic reconstruction interface. In this session, the face of the ST. MATTHEW model (original: 26M points) was refined. In (a), the result of an initial reconstruction has been loaded (time: 4 seconds). In (b), an update region has been selected and the reconstruction is shown refined in (c) and (d) (time: 11 seconds).

(Figs. 1, 15, 16, 17, 18, 19). The LUCY and ST. MATTHEW models were reconstructed using the out-of-core selective algorithm. For both in-core and out-of-core reconstruction, the user must provide a value for the error tolerance ρ_{geom}, which determines the level of detail. An initial selective reconstruction is performed, and the result can then be customized through local update features. We developed a graphic user interface (Fig. 14) to load a reconstructed simplified model from disk and interactively change its level of detail locally using the tool described in [2]. Timings and memory usage for initial selective reconstructions as well as for local updates are reported in Table 1. All the results presented here were obtained on a Pentium IV 3.0GHz, 2GB RAM workstation. These timings include the preprocessing time required to build the kd-tree data structure(s), select the representative and landmark sample points, and construct the initial Delaunay triangulation(s). Table 2 summarizes the overall execution profile for different in-core reconstructions.

Simplification performance The size of the initial simplified models is typically between 1% and 5% of the size of the original point set, which often suffices to preserve the shape of scanned objects at a mid-scale level, and even at fine scale if the point set is very redundant. The method is capable of producing high-quality simplified models directly, without the need of a subsequent mesh fairing step. The majority of the mesh vertices have valences between five and seven, and most facets have good aspect ratios.

Computational performance For in-core selective reconstructions, we set the number of landmark points to 1 for 2k sample points. The preprocessing time was less than 12 seconds in all tests. According to our experiments, the accelerated selective reconstruction method runs up to *20 times faster* than the original one. The computational overhead involved by update operations in the Delaunay triangulation is largely recovered by the reduction of spatial search domains. For out-of-core reconstructions, the LUCY model and the ST. MATTHEW model were split so that each cluster contained fewer than 3.5M points; the resulting number of clusters was 8 and 15 respectively. The clusters were slightly enlarged (2% of the bounding box diagonal) in order to make their boundaries invisible. The number of initial representative points was set to 1 for 1k sample points and the initial reconstruction took less than 10 seconds in both cases. For each part, the refinement then took less than 2.5 minutes.

Execution profiles show that evaluating $\mathbf{P}_{\mathsf{og}}$ and $\mathbf{Q}_{\mathsf{dt}}$ is by far the most costly task in the selective reconstruction algorithm. While we reduced spatial search domains, the overall cost of spatial search queries still remains proportional to the number of facets through which the surface passes, which is the bottleneck of the current method. Memory usage is also relatively high due to the storage of both a kd-tree and a Delaunay triangulation.

In the initial selective reconstruction step, our method runs slower than the surface reconstruction techniques with simplification proposed in [7], and also requires more memory. However, our method can then perform localized updates at interactive rates, while the reconstructions in [7] cannot evolve so easily. Our method does not involve stitching, and our results are guaranteed to be combinatorial manifolds. The dynamic approach is also powerful because it does not require starting from a well-behaved point sample, which is an advantage for out-of-core surface reconstruction, or even for streaming surface reconstruction.

7 Conclusions and future work

We have proposed a data structure with a selective reconstruction algorithm that efficiently reconstructs simplified mesh surfaces from millions of sample points in a dynamic framework. The simplification procedure relies on a local thickness measure that makes sense in both the continuous and the discrete settings, assuming certain sampling conditions on the input point set. The reconstructed surfaces can be dynamically refined or coarsened, benefiting from the same data structure. We have also proposed an out-of-core selective reconstruction algorithm that can be scaled for input point sets that do not fit into memory.

Our method makes dynamic surface reconstruction practicable for large data sets obtained from laser range scanning, which may be an alternative to the standard surface reconstruction-mesh simplification pipeline. The user can also completely customize the reconstruction in order to emphasize particular details. When visualizing a large object, the precision of the reconstruction can be adapted to the viewpoint or to another region of interest at an interactive rate. An efficient dynamic surface reconstruction framework may be also useful for processing point set streams on a network, since it does not require random access to the data.

In the near future, we plan to further improve the computational performance of our accelerated framework by reducing the number of spatial queries. Information on conflicts could be shared between several facets in order to save spatial queries. Another research direction would be to locally relax the global Delaunay property by choosing approximate candidates and repairing errors on the fly when needed.

We also would like to extend the selective reconstruction algorithm to resampling, in order to produce piecewise smooth surfaces from noisy point sets or sets of range images, following Ohtake et al. [7], for example. Finally, we plan to extend the geometric convection algorithm to reconstruct surfaces in a streaming fashion. This would be a way to improve the scalability of the technique and to incorporate the result into an efficient stream processing pipeline for further geometry processing.

8 Acknowledgments

This research is supported by the French Centre National de la Recherche Scientifique (CNRS) and by the Ministère de l'Éducation Nationale, de l'Enseignement Supérieur et de la Recherche. Point set models were provided courtesy of Stanford Scanning Repository, Digital Michelangelo Project (Stanford), and AIM@Shape (IMATI and INRIA).

References

[1] F. Cazals, J. Giesen, Delaunay Triangulation based Surface Reconstruction: Ideas and Al-

In-core reconstructions

Model		Selective reconstruction			Local update		Mem. usage
name	#points	ρ_{geom}	#points	time	#points	time	
BIMBA	1,873,832	0.65	31,643	0:47	–	–	380 MB
		0.8	57,630	1:01	–	–	395 MB
ASIAN DRAGON	3,609,600	0.65	185,504	2:48	177,324	0:14	778 MB
DAVID	3,617,008	0.6	137,025	2:06	174,628	0:28	754 MB
THAI STATUE	5,001,964	0.65	571,600	6:21	–	–	1,367 MB

Out-of-core reconstructions

Model		Selective reconstruction			Local update		Mem. usage
name	#points	ρ_{geom}	#points	time	#points	time	
LUCY	14,027,872	0.85	550,877	17:40	–	–	765 MB
ST. MATTHEW	26,034,562	0.5	116,846	31:12	130,549	0:11	836 MB

Table 1: Performance of our dynamic surface reconstruction framework for various input point sets. Computational timings are given in minutes:seconds and include preprocessing (construction of kd-trees and initial Delaunay triangulations). Memory usage corresponds to the maximum amount of memory used during the reconstructions, in megabytes. All tests were performed on a Pentium IV 3.0GHz, 2GB RAM workstation.

Model name	BIMBA	ASIAN DRAGON	THAI STATUE
ρ_{geom}	0.8	0.65	0.65
Preprocessing	8.7	6.5	4.2
Evaluation of $\mathbf{P}_{og}$ and $\mathbf{Q}_{dt}$	54.9	66.8	74.0
Evaluation of $\mathbf{Q}_{nn}$	30.7	20.2	13.3
Vertex insertion/removal	5.7	6.5	8.5

Table 2: Execution profile for three selective reconstructions. For each model, the column reports the percentages of overall time spent to accomplish the tasks listed on the left.

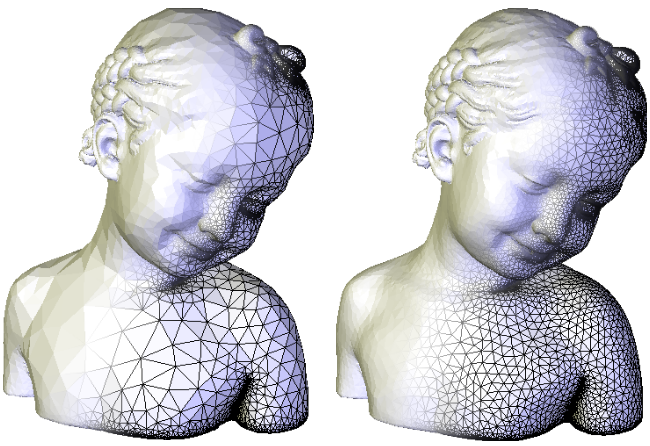

Figure 15: Reconstruction of the BIMBA model (1.9M points) with $\rho_{geom} = 0.65$ (left, 98% of points removed) and $\rho_{geom} = 0.8$ (right, 97% of points removed).

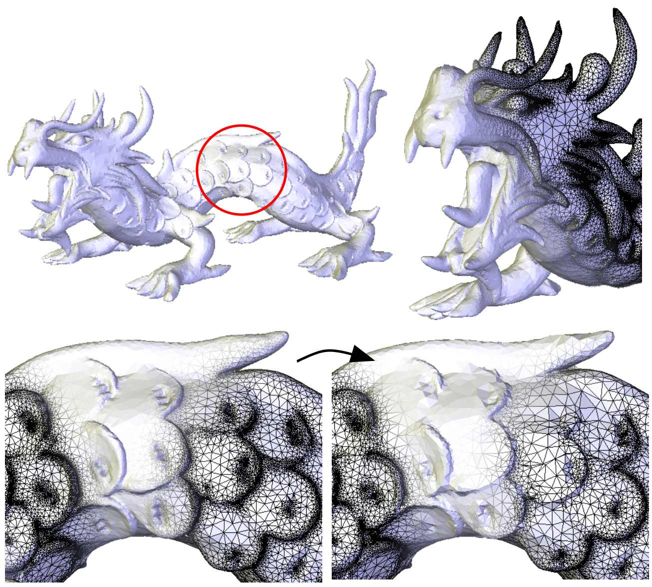

Figure 16: Reconstruction of the ASIAN DRAGON model (3.6M points) with $\rho_{geom} = 0.65$ (95% of points removed). The scales on the back were coarsened in a second step.

gorithms, Tech. Rep. 5393, INRIA (November 2004).

[2] R. Allègre, R. Chaine, S. Akkouche, Convection-Driven Dynamic Surface Reconstruction, in: Proc.

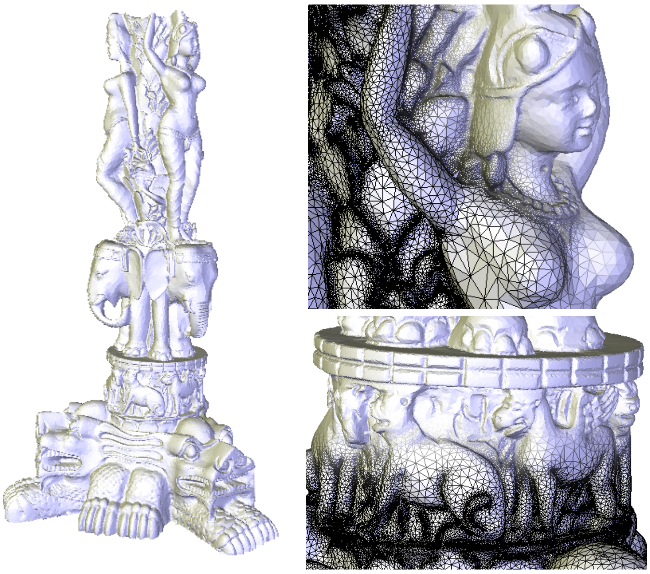

Figure 17: Reconstruction of the THAI STATUE model (5M points) with $\rho_{geom} = 0.65$ (89% of points removed).

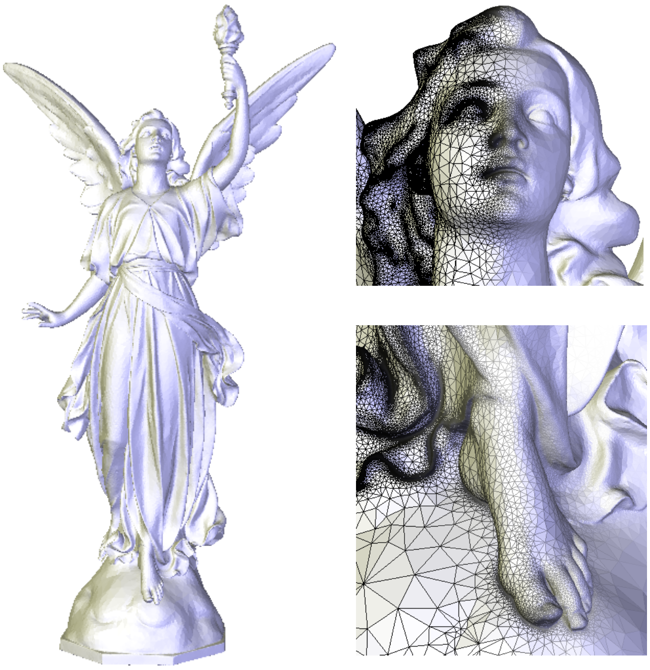

Figure 18: Reconstruction of the LUCY model (14M points), with $\rho_{geom} = 0.85$ (96% of points removed).

Shape Modeling International, IEEE Computer Society Press, 2005, pp. 33–42.

[3] R. Allègre, R. Chaine, S. Akkouche, A Dynamic Surface Reconstruction Framework for Large Point Sets, in: Proc. IEEE/Eurographics Symposium on Point-Based Graphics, 2006, pp. 17–26.

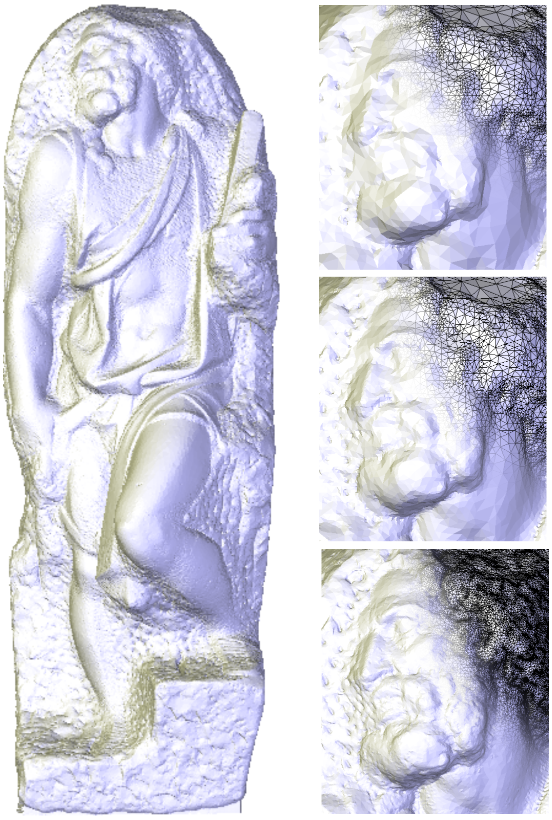

Figure 19: Reconstruction of the ST. MATTHEW model (26M points) at different levels of detail. Left: $\rho_{geom} = 0.8$; right, bottom-up: $\rho_{geom} = 0.8$ (1.2M points), $\rho_{geom} = 0.6$ (180k points), $\rho_{geom} = 0.5$ (117k points).

[4] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, G. Taubin, The Ball-Pivoting Algorithm for Surface Reconstruction, IEEE Transactions on Visualization and Computer Graphics 5 (4) (1999) 349–359.

[5] T. K. Dey, J. Giesen, J. Hudson, Delaunay-based shape reconstruction from large data, in: Proc. IEEE Symposium in Parallel and Large Data Visualization and Graphics, 2001, pp. 19–27.

[6] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, H.-P. Seidel, Multi-level Partition of Unity Implicits, ACM Transactions on Graphics (Proc. SIGGRAPH) 22 (3) (2003) 463–470.

[7] Y. Ohtake, A. G. Belyaev, H.-P. Seidel, An integrating approach to meshing scattered point data, in: Proc. ACM Symposium on Solid and Physical Modeling, 2005, pp. 61–69.

[8] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, D. Fulk, The Digital Michelangelo Project: 3D Scanning of Large Statues, in: Proc. SIGGRAPH, 2000, pp. 131–144.

[9] C. Rocchini, P. Cignoni, F. Ganovelli, C. Montani, P. Pingi, R. Scopigno, The Marching Intersections Algorithm for Merging Range Images, The Visual Computer 20 (2–3) (2004) 149–164.

[10] T. K. Dey, J. Giesen, J. Hudson, Decimating samples for mesh simplification, in: Proc. Canadian Conference on Computational Geometry, 2001, pp. 85–88.

[11] L. Linsen, Point cloud representation, Tech. Rep. 2001-3, Universität Karlsruhe, Germany (2001).

[12] J. Wu, L. P. Kobbelt, Optimized Sub-Sampling of Point Sets for Surface Splatting, in: Proc. Eurographics, 2004, pp. 643–652.

[13] T. K. Dey, J. Giesen, J. Hudson, Sample shuffling for quality hierarchic surface meshing, in: Proc. 10th International Meshing Roundatble Conference, 2001, pp. 143–154.

[14] M. Pauly, M. Gross, L. P. Kobbelt, Efficient Simplification of Point-Sampled Surfaces, in: Proc. IEEE Visualization Conference, 2002, pp. 163–170.

[15] C. Moenning, N. A. Dogson, Intrinsic point cloud simplification, in: Proc. GraphiCon, 2004.

[16] J.-D. Boissonnat, F. Cazals, Coarse-to-fine surface simplification with geometric guarantees, in: Proc. Eurographics, 2001, pp. 490–499.

[17] R. Chaine, A geometric convection approach of 3-D reconstruction, in: Proc. Eurographics Symposium on Geometry Processing, 2003, pp. 218–229.

[18] H. Edelsbrunner, Surface reconstruction by wrapping finite point sets in space, in: B. Aronov, S. Basu, J. Pach, S.-V. M. Sharir (Eds.), Ricky Pollack and Eli Goodman Festscrift, Springer-Verlag, 2002, pp. 379–404.

[19] J. Giesen, M. John, The Flow Complex: A Data Structure for Geometric Modeling, in: Proc. ACM-SIAM Symposium on Discrete Algorithms, 2003, pp. 285–294.

[20] H.-K. Zhao, S. Osher, R. Fedkiw, Fast Surface Reconstruction using the Level Set Method, in: Proc. IEEE Workshop on Variational and Level Set Methods in Computer Vision (VLSM), 2001, pp. 194–202.

[21] R. Allègre, R. Chaine, S. Akkouche, Convection-Driven Dynamic Surface Reconstruction, Tech. Rep. RR-LIRIS-2005-009, LIRIS CNRS - Université Lyon 1, France (January 2005).

[22] N. Amenta, M. Bern, Surface Reconstruction by Voronoi Filtering, in: Proc. Symposium on Computational Geometry, 1998, pp. 39–48.

[23] D. Attali, J.-D. Boissonnat, H. Edelsbrunner, Stability and computation of medial axes: a state of the art report, in: T. Möller, B. Hamann, B. Russell (Eds.), Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration, Springer-Verlag, Mathematics and Visualization, 2007.

[24] F. Chazal, A. Lieutier, The lambda medial axis, Graphical Models 67 (4) (2005) 304–331.

[25] J.-D. Boissonnat, S. Oudot, Provably good sampling and meshing of lipschitz surfaces, in: Proc. Symposium on Computational Geometry, 2006, pp. 337–346.

[26] T. Akenine-Möller, Fast 3D Triangle-Box Overlap Testing, Journal of Graphics Tools 6 (1) (2001) 29–33.

[27] P. Lindstrom, Out-of-core simplification of large polygonal models, in: Proc. SIGGRAPH, 2000, pp. 259–262.

[28] P. Cignoni, C. Montani, C. Rocchini, R. Scopigno, IEEE Transactions on Visualization and Computer Graphics, External Memory Management and Simplification of Huge Meshes 9 (4) (2003) 525–537.

[29] P. Cignoni, F. Ganovelli, E. Gobbetti, F. Marton, F. Ponchio, R. Scopigno, Adaptive TetraPuzzles – Efficient Out-of-core Construction and Visualization of Gigantic Polygonal Models, ACM Transactions on Graphics (Proc. SIGGRAPH) 23 (3) (2004) 796–803.

[30] `http://www.cgal.org`.