Smooth-Surface Reconstruction in Near-Linear Time*

Stefan Funke† Edgar A. Ramos†

Abstract

A surface reconstruction algorithm takes as input a set of sample points from an unknown closed and smooth surface in 3-d space, and produces a piece-wise linear approximation of the surface that contains the sample points. Recently, several algorithms with a correctness guarantee have been proposed. They have unfortunately a worst-case running time that is quadratic in the size of the input because they are based on the construction of 3-d Voronoi diagrams or Delaunay tetrahedrizations which can have quadratic size. In this paper, we describe a new algorithm that also has a correctness guarantee but whose worst-case running time is $O(n \log n)$ where n is the input size. This is actually optimal. As in some of the previous algorithms, the piece-wise linear approximation produced by the new algorithm is a triangulation which is a subset of the 3-d Delaunay tetrahedrization.

1 Introduction

We consider the problem of computing a piecewise linear approximation Σ – more specifically, a triangulation – to a smooth surface S from a subset $P \subset S$ of n points. We refer to P as a *sampling* (set) from S, and to each $p \in P$ as a *sample* (point). This problem has received considerable attention in computer graphics and more recently in computational geometry. In the former area, we only mention the early work by Hoppe *et al* [17] and by Curless and Levoy [10], and more recently, the work of Bernardini *et al.* [5] and of Gopi *et al* [16] (the latter two are closer to the work in computational geometry). These algorithms do not have a *correctness guarantee.* In the latter area, recently, an algorithm (CRUST) with a correctness guarantee under a certain *sampling condition* was proposed by Amenta and Bern [1], and a simplified version of it (COCONE) was described by Amenta *et al.* [2]. Other algorithms have been proposed by Amenta *et al* [3] (POWER CRUST) and Boissonnat and Cazals [7] (interpolation based on natural coordinates). Given a "valid sampling" P, the CRUST and COCONE algorithms output a set of triangles in the Delaunay tetrahedrization of P, which form a surface that approximates and is also topologically equivalent to S. The POWER CRUST algorithm outputs a subset of a power diagram (the dual of a weighted Delaunay triangulation) of the set of points P with some appropriate weights. The algorithm of Boissonat and Cazals uses the Voronoi diagram of the samples to obtain some "natural coordinates" that are then used to obtain the reconstruction, and also outputs a set of triangles in the Delaunay tetrahedrization. Since computing a (weighted) Voronoi diagram or its dual Delaunay tetrahedrization for P requires $\Theta(n^2)$ time in the worst case [13], these previous algorithms have also this quadratic worst case time behavior. On the other hand, since the size of Σ is linear in n, a natural question is whether a surface reconstruction can be computed in near linear time. This is important in practice because current scanning techniques produce a number of samples in the order of several hundred thousands or even up to a million. One can ask whether the worst case quadratic size of Delaunay triangulations can occur for a set of samples from a smooth surface. Recent work of Erickson [15] shows that there are smooth surfaces with uniform sets of samples that have a Delaunay tetrahedrization of quadratic complexity. Therefore, even if the original COCONE algorithm uses an output sensitive algorithm for computing the Delaunay tetrahedrization [9], it could not achieve a subquadratic worst-case running time. In [11], an implementation of the COCONE algorithm was described, which runs in time $O(n \log n)$ if the sampling is "locally uniform". In this paper, we continue with that work and describe a new algorithm that works for any valid sampling and has a correctness guarantee, and whose worst-case running time is near linear. In fact, the running time is optimal: $O(n \log n)$ where n is the input size. As in some of the previous algorithms, the new algorithm outputs a triangulation that is a subcomplex of the 3-d Delaunay tetrahedrization; however, this is obtained by computing only the relevant parts of the 3-d Delaunay structure. The algorithm first estimates for each sample the surface normal and a parameter that is then used to "decimate" the original sampling. The resulting sampling is locally uniform and so a reconstruction based on it can be computed using a simple and fast algorithm as in [11]. For completeness, since [11] is not widely available, we describe here this *basic algorithm.* In a last step, the decimated points are incorporated into the reconstruction. In order to achieve the faster running time, the algorithm uses approximations in the solution of several subtasks. Though the correctness

*Partially supported by the IST Programme of the EU as a Shared-cost RTD (FET Open) Project under Contract No IST-2000-26473 (ECG - Effective Computational Geometry for Curves and Surfaces). An expanded and updated version of this paper can be obtained from `www.mpi-sb.mpg.de/~ramos`

†AG1, Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, Saarbrücken, Germany. E-mail: {`funke,ramos`}`@mpi-sb.mpg.de`.

guarantees (pointwise and normal approximation, topological correctness) of the resulting algorithm are the same as for previous algorithms (*e.g.,* the COCONE), the penalty for using approximations to achieve a faster running time is that some of the constants involved in the correctness statement are worse (for example, the new algorithm is only guaranteed to work for a denser sampling).

Contents. (2) Valid Sampling and Correct Reconstruction; (3) The COCONE Algorithm; (4) Approximation Tools; (5) Locally Uniform Sampling; (6) Basic Algorithm; (7) Outline of the New Algorithm; (8) Estimation; (9) Decimation; (10) Final Reconstruction; (11) Concluding Remarks.

2 Valid Sampling and Correct Reconstruction

We review the framework of "valid sampling" and "correct reconstruction" developed in [1].

Medial Axis and Sampling Condition. The *medial axis* of a surface S in $\mathbb{R}^3$ is the closure of the set of points which have more than one closest point on S. The *local feature size* $\mathrm{lfs}(p)$ at a point $p \in S$ is the distance from p to the medial axis of S. An important property of lfs() is that it is 1-Lipschitz, that is, $\mathrm{lfs}(p) \leq \mathrm{lfs}(q) + \|pq\|$ for any $p, q \in S$. A set P of sample points from S is said to be an *ϵ-sampling* from S if every point $p \in S$ has a sample in P within distance $\epsilon\,\mathrm{lfs}(p)$.

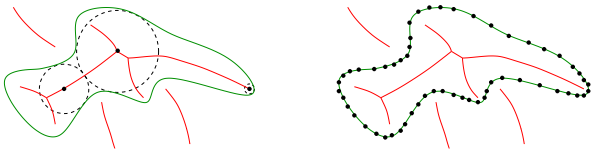

Figure 1: A curve in the plane with its medial axis and a set of samples from the curve. The density of a valid sampling depends on the distance to the medial axis.

Voronoi Diagram and Delaunay Triangulation. The Voronoi diagram $\mathrm{Vor}(P)$ of a set of points P in $\mathbb{R}^3$ is the decomposition of $\mathbb{R}^3$ into (convex) cells of points with the same set of nearest neighbors. Assuming *general position,*, $\mathrm{Vor}(P)$ consists of polyhedrons V_p for $p \in P$, polygons V_{pq} for $p, q \in P$, edges V_{pqr} for $p, q, r \in P$, and vertices V_{pqrs} for $p, q, r, s \in P$ (many of these are empty). The Delaunay tetrahedrization $\mathrm{Del}(P)$ is the dual structure: for each cell V_R where $R \subseteq P$, $\mathrm{Del}(P)$ includes the simplex $\mathrm{conv}(R)$. Thus, $\mathrm{Del}(P)$ consists of tetrahedra, triangles, edges and points.

Restricted Voronoi Diagram and Delaunay Triangulation. The Voronoi diagram $\mathrm{Vor}_{|S}(P)$ of P restricted to S consists of the cells $V_{R|S} = V_R \cap S$ for $R \subseteq P$. It is said to satisfy the *ball* property if each $V_{R|S}$ is a *topological ball:* if V_R is a polyhedron, a polygon, an edge or a vertex then respectively $V_{R|S}$ is topologically equivalent to a polygon, a segment, a point, or empty. Then, the Delaunay triangulation $\mathrm{Del}_{|S}(P)$ of P restricted to S is the structure dual to $\mathrm{Vor}_{|S}(P)$: for each nonempty $V_{R|S}$ in $\mathrm{Vor}_{|S}(P)$, $\mathrm{Del}_{|S}(P)$ contains the simplex $\mathrm{conv}(R)$.

Correct Reconstruction. If P is a "valid sampling" from S then $\mathrm{Vor}_{|S}(P)$ satisfies the ball property and $\mathrm{Del}_{|S}(P)$ is a "correct reconstruction" of S. More precisely, Amenta and Bern [1] have shown that:

THEOREM 2.1. *There exists $\epsilon_0 > 0$ such that for all $\epsilon \leq \epsilon_0$, smooth surface S in $\mathbb{R}^3$ and ϵ-sampling $P \subseteq S$, $\widetilde{S} = \mathrm{Del}_{|S}(P)$ satisfies:*

(i) BIJECTION
$\mu : \widetilde{S} \to S$, determined by closest point, is a bijection.

(ii) POINTWISE APPROXIMATION
For all $x \in S$, $d(x, \mu(x)) = O(\epsilon^2 \mathrm{lfs}(x))$.

(iii) NORMAL APPROXIMATION
For all $x \in S$, $\angle \mathbf{n}_S(x) \mathbf{n}_{\widetilde{S}}(\mu(x)) = O(\epsilon)$, where $\mathbf{n}_F(y)$ is the (outside) normal of F at y.[1]

(iv) TOPOLOGICAL CORRECTNESS
S and $\widetilde{S}$ have the same topological type.

Unfortunately, different surfaces S with the same sampling set $P \subseteq S$ correspond to different triangulations $\mathrm{Del}_{|S}(P)$. That is, we cannot aim to construct $\mathrm{Del}_{|S}(P)$. However, Amenta *et al* [1, 2] have shown that an equally good triangulation can be determined (with possibly worse approximation constants). We describe next their COCONE algorithm.

3 The COCONE Algorithm

For points o and p, let $\overrightarrow{op}$ denote the vector from o to p. Let θ_0 be a parameter with $0 < \theta_0 < \pi/8$. For a sample p with estimated normal $\widetilde{\mathbf{n}}_p$, its *co-cone region* $\text{co-cone}(p, \widetilde{\mathbf{n}}_p, \theta_0)$ is the set of points q for which the angle between $\overrightarrow{pq}$ and $\widetilde{\mathbf{n}}_p$ is in $(\pi/2 - \theta_0, \pi/2 + \theta_0)$. We also denote the co-cone region briefly with C_p, when the other parameters are understood. The algorithm of Amenta *et al.* [2] proceeds in four steps:

1. DELAUNAY TETRAHEDRIZATION. Compute $\mathrm{Del}(P)$.
2. NORMAL ESTIMATION. For each sample $p \in P$ obtain an estimate $\widetilde{\mathbf{n}}_p$ of the surface normal at p as follows: Let $\widetilde{\mathbf{n}}_p$ be $\overrightarrow{pv}$ or $-\overrightarrow{pv}$, where v is the furthest circumcenter over all tetrahedra in $\mathrm{Del}(P)$ incident to p (this includes tetrahedra with a point at infinity). [Dually, v is the vertex of V_p furthest from p.]

[1] For $\widetilde{S}$ the normal is well-defined in the interior of triangles; at edges and vertices, it can be defined as an interpolation from that at the incident triangles.

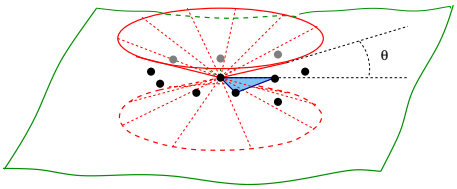

Figure 2: The candidate triangles for p are those Delaunay triangles incident to p that are nearly orthogonal to estimated normal at p and have empty sphere close to equatorial.

3. CANDIDATE TRIANGLES. Put a triangle $t = \triangle p_1 p_2 p_3 \in \text{Del}_{|S}(P)$ in the set T of *candidate triangles*, if t has a circumcenter within C_{p_i} for each p_i, where $C_{p_i} = \text{co-cone}(p_i, \widetilde{\mathbf{n}}_{p_i}, \theta_0)$. [Dually, if there is a $q \in e_t$ within C_{p_i} for each p_i, where e_t is the Voronoi edge dual to t.]
4. SURFACE EXTRACTION. Extract from T the approximating surface Σ.

In Step 2, the sign is determined so that the normals are consistently oriented in all points. This can be achieved by propagating this information through close neighbors. Intuitively, Step 3 selects a Delaunay triangle if (i) its normal is "close" to the estimated normal of its vertices, and (ii) it has a circumsphere that is "not far" from being "equatorial." Step 4 selects from the candidate triangles a subset that makes up a closed surface (see [1] for details). Its success is guaranteed by the fact that $\text{Del}_{|S}(P) \subseteq T$. For this, time linear in the number of candidate triangles suffices. Since this step requires only linear time in the number of candidate triangles, we do not need to worry about the details later when we describe a faster implementation of the COCONE algorithm under a uniformity condition.

Correctness. Amenta *et al.* show that if P is a valid sampling from a smooth surface S, then Σ approximates S; more precisely: Though Σ cannot be guaranteed to be $\text{Del}_{|S}(P)$, it is as good in that it approximates S as stated in the theorem above.

THEOREM 3.1. *There exists $\epsilon_1 \leq \epsilon_0$ such that for all $\epsilon \leq \epsilon_1$, smooth surface S in $\mathbb{R}^3$ and ϵ-sampling $P \subseteq S$, the triangulated surface $\Sigma(P)$ output by the* COCONE *algorithm satisfies the same approximation bounds as in the theorem above (possibly with different constants).*

How small one can choose the parameter θ_0 in the algorithm, depends on ϵ. In the COCONE algorithm θ_0 is set to the value corresponding to the largest ϵ for which the algorithm works.

Worst-Case Running Time. A priori, since the construction of the Delaunay tetrahedrization has a quadratic worst-case running time, so does the COCONE algorithm. A more careful analysis is required since it could be the case that the quadratic behavior do not occur for a set sampling a smooth surface. This has been investigated by Erickson [15]. For example, he describes how the usual quadratic construction consisting of points on two line segments can be "embedded" on a smooth surface. This example also shows that the number of candidate triangles can be quadratic in the worst-case, so it is not feasible to obtain a better algorithm by computing the candidate triangles in an output-sensitive manner, say similar to the method in [9] to compute a Delaunay tetrahedrization. The previous example is however unsatisfactory in that the sampling is highly non-uniform. Erickson provides another construction in which the quadratic construction is part of a locally uniform sampling from a smooth surface.

4 Approximation Tools

In this section, we describe the approximation tools used in the surface reconstruction algorithm. We briefly review the basic tools – balanced decomposition trees and well separated decompositions – and indicate their use to our specific purpose: approximate k nearest neighbor, approximate range reporting, approximate neighborhoods and "Lipschitzation".

4.1 Balanced Decomposition Trees and Applications

Balanced Aspect Ratio (BAR) trees [12] and *Balanced Box Decomposition* (BBD) trees [4] are hierarchical decompositions of the space determined by a set P of n points. They can be efficiently computed in time $O(n \log n)$, require storage $O(n)$, and can be used to answer approximate nearest neighbor and range reporting queries.

Approximate Nearest Neighbors. A *k-th nearest neighbor* (NN) of a point q in P is a point $p \in P$ such that at most $k-1$ other points in P are closer to q than p, and at least $k-1$ other points are no farther from q than p. A *k-th δ_N-approximate nearest neighbor* (δ_N-ANN) of q is a point $p \in P$ for which $d(q,p) \leq (1+\delta_N) \cdot d(q,p')$ where p' is a k-th NN of q; k different points $p_1, p_2, \ldots, p_k \in P$ are *k δ_N-ANNs* of q if $d(q,p_i) \leq (q,p_{i+1})$, and p_i is an i-th δ_N-ANN of q. Note that if $p_1, \ldots, p_k$ are k δ_N-ANNs of q in P with p_k a k-th δ_N-ANN, then all $p \in P$ with $d(q,p) \leq d(q,p_k)/(1+\delta_N)$ are included in $p_1, \ldots, p_{k-1}$.

In our reconstruction algorithm we need a data structure that for a query point q reports k δ_N-ANNs of q in P efficiently. For constant δ_N, data structures presented using BBD [4] and BAR [12] trees can be constructed in time $O(n \log n)$ and they report a set of k δ_N-ANNs in time $O(\log n + k)$. The constant factor in the query time depends on δ_N, but since we do not need to choose δ_N too small, this dependency is not important.

Approximate Range Reporting. In several parts of our

algorithm, given a set of points P in $\mathbb{R}^3$, we need to report those points inside a query sphere. It does not seem possible to implement this query exactly within the time bound we are aiming. Therefore, we use approximate range reporting. This means that for some δ_R, all the points within distance $\widetilde{R}_p$ from p are reported as well as some within distance between $\widetilde{R}_p$ and $(1+\delta_R)\widetilde{R}_p$.

For constant δ_R, the data structures in [4] and [12] also provide an efficient implementation: The query time is $O(\log n + k)$ where k is the number of points reported.

4.2 Well-Separated-Pair Decompositions

Given a set $A \subseteq \mathbb{R}^3$, let size(A) denote the radius of the smallest ball enclosing A (this is not essential, we could also use the smallest enclosing axis-aligned box which is simpler to compute). Given two sets $A, B \in \mathbb{R}^3$, let $\|AB\|$ denote the distance between the centers of their smallest enclosing balls; A and B are said to be *well-separated* if $\|AB\| \geq s \cdot \max\{\text{size}(A), \text{size}(B)\}$, where s is a parameter referred to as the *separation*. The interaction product $A \otimes B$ is the set $\{\{p,q\} : p \in A, q \in B, p \neq q\}$. A *well-separated-pair decomposition* (WSPD) of P is a collection of well-separated pairs $\{A_i, B_i\}$ where $A_i, B_i \subseteq P$ with each $\{A_i, B_i\}$ a well-separated pair, and with $P \otimes P = \bigcup_{i=1}^{k} A_i \otimes B_i$ as a disjoint union. Callahan and Kosaraju (1995) have shown that a WSPD of size $O(s^3 n)$ can be constructed in time $O(n \log n + s^3 n)$. Their construction is based on a tree decomposition $T = T(P)$ of the point set: each node ν of the tree is associated with a subset A_ν of P so that the root associated with P and the n leaves associated with the singleton sets; the sets associated with the left and right children, ν_l and ν_r, of a node ν are obtained by splitting the smallest enclosing rectangle of A_ν with a plane parallel to its longest side at its middle point. We call the sets A_ν *clusters*. Their WSPD consists of well-separated *cluster-pairs* $\{A_i, B_i\}$ where A_i and B_i are clusters in T. We omit a description of how the decomposition is obtained. If $\{A_i, B_i\}$ is in the WSPD, we say that A_i and B_i are *direct partners.* For our purposes it suffices to be familiar with the structure of the resulting construction: Each cluster A_ν in T has associated a set $\mathcal{B}_\nu$ of direct partners $\mathcal{B}_\nu = \{B_{\nu,1}, B_{\nu,2}, \ldots, B_{\nu,k_\nu}\}$. In particular, it is important that all the interactions of a point $p \in P$ with the other points $P - \{p\}$ can be captured by traversing the path in T from the root to the leaf p. For a node ν in T, let $L(\nu)$ be the set consisting of ν and its (proper) ancestors in T, and let

$$\mathcal{B}_\nu^+ = \bigcup_{\mu \in L(\nu)} \mathcal{B}_\mu,$$

the set of *inherited partners* of A_ν (inherited from its ancestors in T). Also let $\mathcal{B}_\nu^* = \mathcal{B}_\nu^+ \cup \mathcal{B}_\nu$, the set of all partners of A_ν. Thus, the set $\mathcal{B}_p^* = \mathcal{B}_{\{p\}}^*$ of partners of p cover all the interactions of p.

We will be interested in computing for every $p \in P$ some quantity $M(p) = M(\{p\} \otimes (P - \{p\}))$ that depends on the interaction of p with all the other points. In general, this computation would require quadratic time, so we will specifically consider an approximate version $\widetilde{M}(p) = \widetilde{M}(p, \mathcal{B}_p^*)$ that depends on the interactions of p with all its well-separated partners. This approximate version can be efficiently computed under the assumptions:

(i) for each well-separated pair A, B, $\widetilde{M}(A, \{B\})$ can be computed in time $O(1)$ [base case];

(ii) $\widetilde{M}(A, \mathcal{B})$ and $\widetilde{M}(A, \mathcal{B}')$ can be combined into $\widetilde{M}(A, \mathcal{B} \cup \mathcal{B}'))$ in time $O(1)$ [merging];

(iii) for $A' \subseteq A$, $\widetilde{M}(A, \mathcal{B}) = \widetilde{M}(A', \mathcal{B})$ [inheritance].

Then the $\widetilde{M}(p)$'s can be computed using linear time by performing a top-down traversal of T that computes $\widetilde{M}(A_\nu, \mathcal{B}_\nu^*)$ for each ν: when visiting ν, it computes $\widetilde{M}(A_\nu, \mathcal{B}_\nu)$, in time $O(|\mathcal{B}_\nu|)$, making use of assumptions (i) and (ii), and then combines it with $\widetilde{M}(A_{\text{parent}(\nu)}, \mathcal{B}_{\text{parent}(\nu)}^*)$ to obtain $\widetilde{M}(A_\nu, \mathcal{B}_\nu^*)$, in time $O(1)$, making use of assumption (iii). Thus, the total time is $O(\sum_\nu |\mathcal{B}_\nu|)$, which is $O(n)$.

We will use this computation scheme several times, and we will refer to it as a WSPD-computation. For example, let us consider $M(p)$ to be a δ_W-approximate closest neighbor of p in $P - \{p\}$. We take $s = \Theta(1/\delta_W)$ so that $(1-\delta_W)\|AB\| \leq \|pq\| \leq (1+\delta_W)\|AB\|$, for a well-balanced pair A, B and $p \in A, q \in B$. Then the base case is $M(A, \{B\}) = \|AB\|$; merging is simply taking the closest and inheritance is clear.

4.3 Approximate Neighborhoods

In our estimation, we will use the concept of *approximate neighborhood* (AN) of a point. For a fixed angle ϕ, an *approximate neighborhood* of p is a subset $\text{AN}(p) = \text{AN}_\phi(p)$ of P such that for any $q \in P - \{p\}$ there is $q' \in \text{AN}(p)$ such that $\angle qpq' \leq \phi$ and $\|pq\| \geq (1-\delta_\phi)\|pq'\|$ where $\delta_\phi = \sin\phi$ (see Fig. 3). In general, it is not necessary that $\text{AN}(p) \subseteq P$. An AN can be obtained by covering the space of directions with a set of cones of angle ϕ,[2] a *ϕ-cone covering* of S^2 for short, and then selecting in each an approximate closest point. Clearly, $\Omega((\pi/\phi)^2)$ cones of angle ϕ are necessary to cover the space of directions.

AN's can be obtained through a WSPD-computation using overall time $O(n \log n)$. First, let C_ϕ be a ϕ-cone covering of S^2. For each split-tree set A, let $r(A)$ be a representative point ($r(A)$ may be a point not in A, *e.g.,* the center of the smallest enclosing sphere). For each split-tree set A, we want to compute an approximate neighborhood $\text{AN}(A_\nu, \mathcal{B}_\nu^*) =$

[2] More precisely, a cone with apex c, axis $\overrightarrow{v}$ and angle ϕ is the set of all points p such that the angle between the vector from c to p and $\overrightarrow{v}$ is no greater than ϕ.

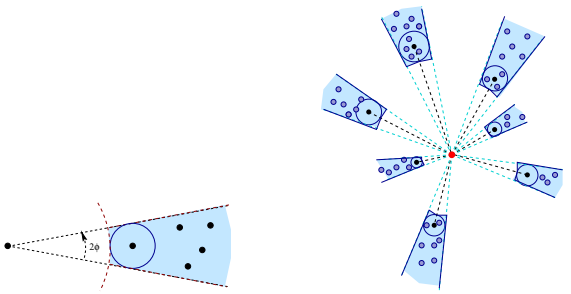

Figure 3: Approximate neighborhood of a sample point.

$\{(c, p_c) : c \in C_\phi\}$, where p_c is an "approximate neighbor" of A in the cone c (but it may be outside of it). In particular, $\text{AN}(\{p\}, \mathcal{B}^*_{\{p\}})$ will be an AN of P. We only need to specify (i) and (ii) in the WSPD-computation scheme:

(i) For a well-separated pair A, B, $\text{AN}(A, \{B\})$ consists of $(c, r(B))$ where $c \in C_\phi$ is the cone containing the vector from $r(A)$ to $r(B)$, and (c, ∞) for all other c;

(ii) given $\text{AN}(A, \mathcal{B})$ and $\text{AN}(A, \mathcal{B}')$, they are merged into the set $\text{AN}(A, \mathcal{B} \cup \mathcal{B})$ consisting of the pairs (c, p) such that $(c, q) \in \text{AN}(A, \mathcal{B})$, $(c, q') \in \text{AN}(A, \mathcal{B}')$, and p is the closest to $r(A)$ among q and q'.

Clearly, both operations can be performed in time $O(1)$ (actually in time $O(|C_\phi|) = O((\pi/\phi)^2)$), assuming that the representatives $r(A)$ are available. Because of the approximations involved the choice of a representative and the inheritance in the WSPD-computation, the result is not a ϕ-AN, rather a $K\phi$-AN for some constant $K \geq 1$. With an arbitrary point of A as $r(A)$, $K \approx 4$, see Fig. 4. A better K is possible if $r(A)$ is allowed to be other than a point in A. For example, if $r(A)$ is the center of the smallest enclosing ball, and for a cone c with apex at p, p_c in (c, p_c) is taken to be on the axis of c and at the same distance to p as the closest representative in the cone; then $K \approx 2$. Different variants of the method and better analysis should be investigated to improve the value of this constant. We will assume that $r(A) \in A$, so that $AN(p) \subseteq P \setminus \{p\}$, but this is not essential.

In the subsequent presentation, we use ϕ_0 for the angle used in the approximate neighborhood (recall that θ_0 is the parameter in the COCONE algorithm),

4.4 "Lipschitzation"

Our definition of local uniform sample involves an α-*Lipschitz* function: For $\alpha > 0$, a function $f : S \to \mathbb{R}^+$ has the α-Lipschitz property if for any $x, y \in S$, $f(x) \leq f(y) + \alpha\|xy\|$. Given an arbitrary function $g : P \to \mathbb{R}^+$, there are two "natural" ways to extend it to S and make it α-Lipschitz; they correspond to whether small or large values

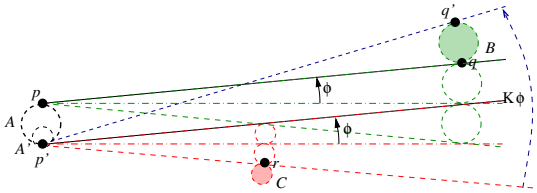

Figure 4: $\{A, B\}$ and $\{A', C\}$ are direct cluster-pairs; A' is a cluster contained in A and so inherits B as a partner. The points p, p', q, r are the representatives of the clusters A, A', B, C respectively. The cones with angle ϕ and apex p and p' are corresponding ones in the cone covering at p and at p'. The point q' in B is within a cone with apex at p', axis $p'r$ and angle approximately 4ϕ.

at points in P dominate:

$$\begin{aligned} \mathrm{L}_m f(x) &= \min_{p \in P}\{f(p) + \alpha\|xp\|\}; \\ \mathrm{L}_M f(x) &= \max_{p \in P}\{f(p) - \alpha\|xp\|\}. \end{aligned}$$

A simple calculation shows that these functions are indeed α-Lipschitz.

Though $\mathrm{L}_m f$ and $\mathrm{L}_M f$ are defined everywhere on S (this is needed in the analysis), we are interested in computing them only for points in P. Unfortunately, a exact computation of them requires quadratic time. So, we describe here approximate versions that will suffice in our application and can be computed in time $O(n)$ (given a WSPD). First, we say that a function $f : X \to \mathbb{R}$ is δ-approximately α-Lipschitz if for all $x, y \in X$, $f(y) \leq (1 + \delta)(f(x) + \alpha\|xy\|)$. The approximate versions, denoted $\widetilde{\mathrm{L}}_m$ and $\widetilde{\mathrm{L}}_M$, will be defined by their implementation through the WSPD-computation approach. Let $f : P \to \mathbb{R}^+$ be given. Then we can define f_m and f_M on subsets of P as $f_m(A) = \min_{p \in A} f(p)$ and $f_M(A) = \max_{p \in A} f(p)$. The basis of the computation is $\widetilde{\mathrm{L}}_m f(A, \{B\}) = f_m(B) + \alpha\|AB\|$, merging is defined by $\widetilde{\mathrm{L}}_m f(A, \mathcal{B} \cup \mathcal{B}') = \min\{\widetilde{\mathrm{L}}_m f(A, \mathcal{B}), \widetilde{\mathrm{L}}_m f(A, \mathcal{B}')\}$. This defines $\widetilde{\mathrm{L}}_m f$ for $p \in P$; for $x \notin P$, we define $\widetilde{\mathrm{L}}_m f(x) = \mathrm{L}_m f(x)$. In what follows, let δ_W be the approximation error introduced by the WSPD-computation. We have that $\widetilde{\mathrm{L}}_m$ is a "good approximation" to L_m and is approximately α-Lipschitz:

LEMMA 4.1. *If $f : P \to \mathbb{R}$ with $P \subset S$, then:*

(i) For $x \in S$,

$$(1 - \delta_W)\mathrm{L}_m f(x) \leq \widetilde{\mathrm{L}}_m f(x) \leq (1 + \delta_W)\mathrm{L}_m f(x);$$

(ii) For $x, y \in S$,

$$\widetilde{\mathrm{L}}_m f(x) \leq (1 + 3\delta_W)(\widetilde{\mathrm{L}}_m f(y) + \alpha\|xy\|).$$

For our application, L_M rather than L_m is the relevant operator. Unfortunately, the approximate operator $\widetilde{\mathrm{L}}^I_M$ obtained analogously to $\widetilde{\mathrm{L}}_m$, but substituting max for min, does not lead to the desired properties: the resulting function may not be approximately α-Lipschitz. This is a result of the substraction in the definition. We define then $\widetilde{\mathrm{L}}_M$ as $\widetilde{\mathrm{L}}_m\widetilde{\mathrm{L}}^I_{M|P}$, that is, first apply $\widetilde{\mathrm{L}}^I_M$ and then apply $\widetilde{\mathrm{L}}_m$ to the resulting function restricted to P. $\widetilde{\mathrm{L}}_M f$ is certainly $(1+3\delta_W)$-approximately Lipschitz, since it uses $\widetilde{\mathrm{L}}_m$ as the last operation in its definition. It remains to verify that $\widetilde{\mathrm{L}}_M$ is a "good approximation" to L_M and is approximately α-Lipschitz:

LEMMA 4.2. *If $f : P \to \mathbb{R}$ with $P \subset S$, then:*

(i) For $x \in S$ and for $q \in P$, respectively,

$$\begin{aligned}\widetilde{\mathrm{L}}_M f(x) &\geq \max_{p\in P}((1-2\delta_W)f(p) - (1-\delta_W)\alpha\|xp\|)\,;\\ \widetilde{\mathrm{L}}_M f(q) &\leq \max_{p\in P}(f(p) - (1-\delta_W)\alpha\|pq\|)\,.\end{aligned}$$

(ii) For $x, y \in S$,

$$\widetilde{\mathrm{L}}_M f(x) \leq (1+3\delta_W)(\widetilde{\mathrm{L}}_M f(y) + \alpha\|xy\|).$$

5 Locally Uniform Sampling

The new reconstruction algorithm takes advantage of the fact that it is easier to compute a reconstruction for a "locally uniform" set of samples. The concept of local uniformity used in [11] is not sufficient for our purposes. We will not assume that the input sampling is locally uniform; rather, it will be enforced algorithmically through decimation, and for this we need an appropriate definition (which in the end more or less implies that in [11]).

The ϵ-sampling condition imposes an upper bound on the sampling density. A possible uniformity definition is also impose a lower bound by requiring that the samples are not too close to each other:

TIGHT ϵ-SAMPLING: For a constant fraction $\beta > 0$, a sample $P \subset S$ is a *tight ϵ-sampling* if

(i) for all $x \in S$, $B(x, \epsilon\ \mathrm{lfs}(x)) \cap P \neq \emptyset$

(ii) for all $p \in P$, $B(p, \beta(\epsilon\ \mathrm{lfs}(p))) \cap P = \{p\}$.

Since we want the reconstruction algorithm to enforce the uniformity condition on the sample (by decimating), to obtain a tight ϵ-sampling one would have to estimate $\mathrm{lfs}(p)$, for $p \in P$. Unfortunately, at this point, we do not know how to do this efficiently. Furthermore, the algorithm would have to know the value of ϵ. So, we have an alternative definition which is appropriate for our algorithm.

LOCALLY UNIFORM ϵ-SAMPLING: A sample $P \subseteq S$ is a *locally uniform ϵ-sampling,* if there is a *control* function $\varphi : S \to \mathbb{R}^+$ and constants $0 < \alpha, \beta, \delta < 1$, such that:

(i) φ is δ-approximate α-Lipschitz

(ii) For all $x \in S$, $B(x, \varphi(x)) \cap P \neq \emptyset$

(iii) For all $p \in P$, $B(p, \beta\varphi(p)) \cap P = \{p\}$

(iv) For all $x \in S$, $\varphi(x) \leq \epsilon\ \mathrm{lfs}(x)$

Intuitively, the function $\varphi(x)$ controls the local density of samples: it upper bounds the largest empty ball around x, and multiplied by the factor β, it lower bounds the largest empty ball around a sample p. A tight ϵ-sampling is also a locally uniform ϵ-sampling with appropriate constants. A locally uniform sample has several useful properties.

LEMMA 5.1. *For parameters $0 < \epsilon, \alpha, \beta, \delta_U < 1$, and constants $c_E \leq 1/2\alpha$ and $c_V \leq 1/2\alpha - 1$, there are constants c_D, m_E, m_V such that the following holds for any smooth surface S and for any locally uniform ϵ-sampling P from S, with parameters α, β, δ_U:*

(i) BOUNDED-RATIO PROPERTY
For each $p \in P$, if q and q' are a nearest and a furthest neighbor of p in $\mathrm{Del}_{|S}(P)$, then $\|pq'\| \leq c_D\|pq\|$.

(ii) BOUNDED-INCREASE PROPERTY
For each $x \in S$, $B(x,r) \cap P = \emptyset$ implies $|B(x, c_E r) \cap P| \leq m_E$.

(iii) SMALL-SIZE-NEIGHBORHOOD PROPERTY
For each $p \in P$, $|B(p, c_V r_p) \cap P| \leq m_V$, where r_p the radius *of $V_{p|S}$.*

The proof of the Bounded-Ratio Property also implies that the degree of each sample in $\mathrm{Del}_{|S}(P)$ is bounded by a constant, and that the triangles in $\mathrm{Del}_{|S}(P)$ are well-shaped. The Small-Size-Neighborhood Property will be used to argue that the Basic Algorithm (see Section 6) runs in time $O(n \log n)$ (it implies that the Voronoi cell of a sample p inside the co-cone can be computed considering only a constant number of neighbors of p).

6 Basic Algorithm

We assume that a locally uniform ϵ-sampling $P \subseteq S$ is given, and that an estimate of the surface normal $\widetilde{\mathbf{n}}_p$ at each $p \in P$ is already available. We describe an implementation of the COCONE algorithm that under these assumptions run in time $O(n \log n)$ time. For each sample p, it determines successively a larger neighborhood of p until all the samples necessary to determine the portion of V_p inside the co-cone are collected. The correctness of this algorithm follows directly from the correctness of the general COCONE algorithm. It has the advantage that it is still correct even when the sample set is not locally uniform (though then the running time is not longer $O(n \log n)$).
Recall that for a sample p with estimated normal $\widetilde{\mathbf{n}}_p$, its *co-cone region* $C_p = \text{co-cone}(p, \widetilde{\mathbf{n}}_p, \theta_0)$ is the set of points q for

which the angle between $\overrightarrow{pq}$ and $\widetilde{\mathbf{n}}_p$ is in $(\pi/2-\theta_0, \pi/2+\theta_0)$, where θ_0 is a constant angle (derived from the analysis). We describe an alternative implementation of Step 3 in the COCONE algorithm. The algorithm assumes the availability of a data structure for reporting δ_R-ANNs in P (see Section 4).

Algorithm. We only need to describe how to compute the candidate triangles (Step 3); the surface extraction (Step 4) is as in the original COCONE algorithm and we do not need to elaborate. For each $p \in P$, starting with say $c = 6$, repeat the following:

1. $N_p \leftarrow$ set of c δ_R-ANNs of p
2. Compute the Voronoi cell V_p of p with respect to all samples in N_p.
3. Let $d_{\max}$ be the maximum distance from p to points on Voronoi edges of V_p within C_p.
4. if $\max_{q \in N_p} d(p,q) > 3(1+\delta_R) d_{\max}$ then exit
5. $c \leftarrow 2 \cdot c$

Then, output a triangle t as candidate if its dual edge appears in V_p within C_p for each vertex p of t.

Correctness. The algorithm makes sure that the co-cone region of each Voronoi cell (with respect to the estimated normal) is computed exactly: further nearest neighbors are added until one can be sure that the co-cone region is not affected anymore. Therefore, the correctness argument in [2] applies as well.

Running Time. It is easy to see that, without the uniformity assumption on the sampling, the algorithm has a worst-case running time $O(n^2 \log n)$. Under the uniformity assumption, the running time is considerably improved.

Let $p \in P$ and let r'_p be the distance to a furthest point q' on an edge of V_p inside C_p. The algorithm stops when it ensures that all samples yet to be considered are at distance at least $3r'_p$, and so they cannot affect the Voronoi diagram inside the co-cone. We verify that the Bounded-Increase Property implies that at most a constant number of points are considered in computing the co-cone of p.

LEMMA 6.1. *There are constants α and m_V such that the following holds: Let P be a locally uniform sampling, and for $p \in P$ let r'_p the furthest distance from p to an edge of V_p within the co-cone region C_p. Then the number of samples which have distance less than $3r'_p(1+\delta_R)$ from p is at most m_C.*

The particular use of c_V in the proof, determines a constraint on α (since $c_V \leq 1/2\alpha - 1$) and hence on ϵ for which the algorithm works. We will see later that $\alpha \geq K'\epsilon$, where K' is a constant, and in a limit case (when certain δ's become zero) this becomes $\alpha \geq 3\epsilon$. In this latter case, we obtain the constraint $\epsilon \leq 1/6(c_V + 1)$. For example, for ϵ_0 so that $\Delta \leq 1/6$ and with $\delta_R \leq 1/6$, we have $c_V = 5.5$ and $\alpha \leq 1/12$ and $\epsilon \leq 1/40$.

We conclude that under these conditions the cost of computing the co-cone for each sample is $O(\log n)$, and so the total construction time is $O(n \log n)$.

THEOREM 6.1. *Assuming that P is a locally uniform $\epsilon-$sampling of the surface S (with appropriate parameters as discussed above), all candidate triangles can be determined in time $O(n \log n)$, and so the modified* COCONE *algorithm runs in time $O(n \log n)$.*

7 Outline of the New Algorithm

The new reconstruction algorithm follows an approach that we call "Decimate-and-Conquer." The input sampling is "decimated" to obtain a subsampling that is locally uniform and on which the Basic Algorithm efficiently produces a reconstruction. Finally, the decimated samples are introduced into the final reconstruction.

Input: A finite sample set P from an unknown surface S in 3-d space.
Output: A triangulation Σ on P that approximates S.

1. ESTIMATION
   For each sample $p \in P$ estimate the surface normal $\widetilde{\mathbf{n}}_p$ and the decimation radius $\widetilde{R}_p$.
2. DECIMATION
   Obtain a locally uniform subsample $Q \subseteq P$ (using $\widetilde{R}_p$).
3. INITIAL RECONSTRUCTION (Conquer)
   Obtain a reconstruction Σ' based on Q using a Basic Algorithm.
4. FINAL RECONSTRUCTION (Consolidate)
   Obtain a reconstruction Σ based on P as $\mathrm{Del}_{|\Sigma'}(P)$.

The Basic Algorithm used in Step 3 has already been described. In the next three sections, we describe each of the remaining three steps.

8 Estimation

The Estimation Step obtains an estimate of the surface normal at each sample point, and an appropriate control function, the *decimation radius,* which is then used in the next step to decimate the input sampling.

8.1 Normal

The surface normal at $p \in P$ is estimated as follows:

1. Let q_1 be the point in $\mathrm{AN}_{\phi_0}(p)$ nearest to p.
2. Let q_2 be the point in $\mathrm{AN}_{\phi_0}(p)$ that is nearest to p among those that form with q_1 and with respect to p

an angle greater than γ_N, where γ_N is an absolute constant.

3. Let $\widetilde{\mathbf{n}}_p$ be the normal to the triangle $\triangle pq_1q_2$ (with appropriate orientation).

Again here, the appropriate orientation is chosen so that it is consistent over the surface.

Correctness. The two vectors $\overrightarrow{pp_1}$ and $\overrightarrow{pp_2}$ are approximately orthogonal to the normal at p, as guaranteed by the following lemma from [2]:

LEMMA 8.1. *Let $x, x' \in S$ with $|xx'| \leq c\epsilon \cdot \mathrm{lfs}(x)$, $c \leq \sqrt{2}$. Then $|\angle \overrightarrow{xx'} \mathbf{n}_x - \pi/2| \leq \sin^{-1}(c\epsilon/2)$.*

Therefore, since the angle between these vectors is not too small, the normal to the plane they determine is an approximation of the normal at p:

LEMMA 8.2. *If P is an ϵ-sampling from S,then $\angle \mathbf{n}_p \widetilde{\mathbf{n}}_p = O(\epsilon)$.*

Let $\widetilde{T}_p$ the plane through p that is normal to $\widetilde{\mathbf{n}}_p$, that is, $\widetilde{T}_p$ is the estimated surface tangent plane at p.

8.2 Decimation Radius

Intuition. Let r_p be the radius of the smallest ball centered at p that contains $V_{p|S}$ (*i.e.,* the distance to the furthest restricted Voronoi vertex). Intuitively, in decimating P, it is safe to keep $p \in P$ and delete all the other samples

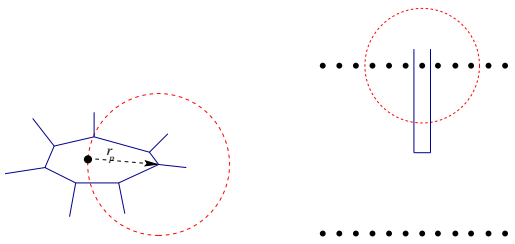

Figure 5: Assuming the sampling condition holds, the radius of the restricted Voronoi cell r_p gives information (a lower bound) on the local feature size (more precisely on ϵ lfs). On the right, we can see that many "redundant" samples are eliminated if one decimates the samples inside the ball centered at p and with radius proportional to r_p.

within a radius proportional to r_p. However, in order to obtain a graded sample set, the decimation radius should be α-*Lipschitz* for an appropriate $\alpha > 0$ (see Section 4). So, let us then define $\varphi_0 : P \to \mathbb{R}^+$ as $\varphi_0(p) = (1+\alpha)r_p$, and $\varphi_1 : S \to \mathbb{R}^+$ as $\mathrm{L}_M\varphi_0$. We claim that φ_1 has the following properties, for $\epsilon \leq 1/3$, $\alpha \geq 3\epsilon$: For each $x \in S$, φ_1 is α-Lipschitz; $B(x, \varphi_1(x)) \cap P \neq \emptyset$; and $\varphi_1(x) \leq (1+2\alpha)\epsilon\ \mathrm{lfs}(x)$. Thus, φ_1 is a lower bound for ϵ lfs and captures the requirement that there must be a sample nearby.

Actual Implementation. Certainly, we cannot determine r_p as we do not even know S. Instead, we obtain a related value $\widetilde{r}_p$ that will serve the same purpose and that can be computed for all samples within $O(n \log n)$ time. Let $V_p = V_p(P)$ be the Voronoi cell of p with respect to P, and $V_{p|S}$ be its restriction to S.

For each p, we determine a point $\widetilde{w}_p$ (not necessarily on S) and its distance $\widetilde{r}_p$ to p such that

(R1) $B(p, (1+\delta_E)\widetilde{r}_p)$ contains $V_{p|S}$, and

(R2) there is a point $w_p \in S$ "close" to $\widetilde{w}_p$ with

$$B(w_p, (1-\delta_E)\widetilde{r}_p) \cap P = \emptyset,$$

for a constant δ_E that depends on θ_0 and ϕ_0. The point $\widetilde{w}_p$ is determined as follows:

1. Let $\widetilde{V}_p$ be the Voronoi cell of p with respect to $\mathrm{AN}_{\phi_0}(p)$ restricted to $\widetilde{T}_p$.
2. Let $\widetilde{w}_p$ be the vertex of $\widetilde{V}_p$ furthest from p and let $\widetilde{r}_p$ be $\|p\widetilde{w}_p\|$.

To see that (R1) holds, note that since $B(p, \widetilde{r}_p)$ contains $\widetilde{V}_p$, then an expanded ball $B(p, (1+\delta_E)\widetilde{r}_p)$ includes all of $V_p \cap C_p$ (the co-cone region of p), where δ_E is an appropriate fraction that depends on θ_0. Since $V_{p|S} \subseteq V_p \cap C_p$ then $B(p, (1+\delta_E)\widetilde{r}_p)$ contains $V_{p|S}$. Fig. 6 illustrates why (R2) holds: the ball $B(\widetilde{w}_p, \widetilde{r}_p)$ is empty of approximate neighbor points; then the shrunk ball $B(\widetilde{w}_p, (1-\delta'_E)\widetilde{r}_p)$ is empty of all samples, where δ'_E is again an appropriate fraction that depends on θ_0 and ϕ_0; then the point w_p on S closest to $\widetilde{w}_p$ satisfies $B(w_p, (1-\delta_E)\widetilde{r}_p) \cap P = \emptyset$ where δ_E is a constant slightly larger than δ'_E.

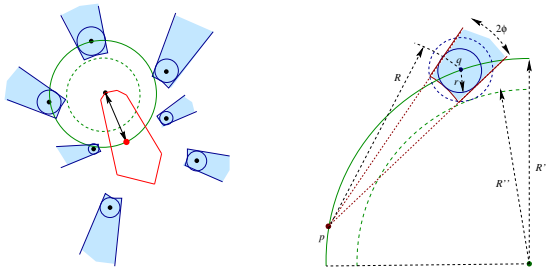

Figure 6: Estimation of furthest restricted Voronoi vertex. The diagram on the right allows to obtain a bound on the shrinking δ'_E that a Voronoi sphere with respect to N_p needs to undergo in order to be empty of points in P. Clearly, δ'_E decreases as ϕ_0 decreases.

Now, we can define the approximate versions of φ_0 and φ_1. Let δ_0 be a constant whose value will depend on the other δ parameters. For $p \in P$ let

$$\widetilde{\varphi}_0(p) = (1+\alpha)(1+\delta_0)\widetilde{r}_p$$

and then let

$$\widetilde{\varphi}_1 = \widetilde{\mathrm{L}}_M \widetilde{\varphi}_0.$$

Now, we verify that $\widetilde{\varphi}_1$ has the desired properties to play the role of a control function.

LEMMA 8.3. *Let*

$$\begin{aligned} K &= ((1+\alpha)(1+\delta_0)+\alpha(1-\delta_W))(1+2\delta_E) \\ c_U &= (1+3\delta_W)(K(1+\epsilon)+\alpha). \end{aligned}$$

with $\delta_0 \geq \delta_E + 2\delta_W$ and α so that $K\epsilon \leq \alpha(1-\delta_W)$, and let $\delta_U = 3\delta_W$. Then $\widetilde{\varphi}_1$ has the following properties:

(D1) $\widetilde{\varphi}_1$ is δ_U-approximately α-Lipschitz.

(D2) For all $x \in S$, $B(x, \widetilde{\varphi}_1(x)) \cap P \neq \emptyset$.

(D3) For all $x \in S$, $\widetilde{\varphi}_1(x) \leq c_U \epsilon \mathrm{lfs}(x)$.

In the case that all δ's are zero, we obtain $K = 1+2\alpha$. Then, to satisfy the constraint $\alpha > K\epsilon$, it is sufficient that $\alpha \geq 3\epsilon$.

Let $\widetilde{R}_p = \beta\widetilde{\varphi}_1(p)$ with $0 < \beta < 1$ be the *decimation radius.*

9 Decimation

The Decimation Step uses $\widetilde{R}_p$ to eliminate some sample points and obtain $Q \subseteq P$ that is locally uniform. It works as follows:

1. $Q \leftarrow \emptyset$, $P' \leftarrow P$
2. Repeat
   (a) $p \leftarrow$ next from P'
   (b) $Q \leftarrow Q \cup \{p\}$
   (c) $P' \leftarrow P' - B(p, \widetilde{R}_p)$
3. until $P' = \emptyset$

The range reporting in Step 2(c) is implemented approximately (see Sec.4) with error parameter δ_R. With this, the total time is proportional to $O(n \log n + \sum_{q \in Q} k_q)$ where k_q is the number of points reported when q is considered. The analysis later will show that $\sum_q k_q = O(n)$ (the uniformity of Q implies that each point of P is reported only a constant number times; thus we do not need a dynamic data structure). For $q \in Q$, let D_q be the set of points decimated by q. We use $\widetilde{R}_p = \beta\widetilde{\varphi}_1(p)$ with $0 < \beta < 1$.

LEMMA 9.1. *The decimated sampling Q has the following properties:*

(D4) For $q \in Q$, $B(q, \beta\widetilde{\varphi}_1(q)) \cap Q = \{q\}$.

(D5) For $x \in S$, $B(x, \gamma\widetilde{\varphi}_1(x)) \cap Q \neq \emptyset$ where

$$\gamma = \frac{1+\beta(1+\delta_R)(1+\delta_U)}{1-\alpha\beta(1+\delta_R)(1+\delta_U)}.$$

(D6) Q is an $(c_S\epsilon)$-sampling where $c_S = c_U\gamma$.

If all the δ's are zero, then we have $\gamma = (1+\beta)/(1-\alpha\beta)$ and $c_S = (1+2\alpha)(1+\beta)/(1-\alpha\beta)$. As an illustration, let $\alpha = 1/6$ and $\beta = 1$. Then $\epsilon \leq 1/18 = 0.056...$, $\gamma = 12/5$, $c_D = 8$, and $c_S = 16/5 = 3.2$. Thus, in decimating, we lose about a factor of 3 in the "quality" of the sample. For a fixed α, there is a trade-off between c_R and c_S depending on the choice of β. As another example, let $\alpha = 1/6$ and $\beta = 1/4$, then $c_E = 30/23 = 1.30...$, $c_R = 120/9 = 13.33...$, and $c_S = 40/23 = 1.739...$ In the limit as $\beta \to 0$, $c_S \to 1+2\alpha$. One can also vary α as long as $3\epsilon \leq \alpha$; in the limit $\epsilon \to 0$, $c_S \to 1$.

10 Final Reconstruction

The last step computes $\Sigma = \mathrm{Del}_{|\Sigma'}(P)$ where Σ' is the initial reconstruction. In analogy with $\mathrm{Del}_{|S}(P)$, which we know it is a correct reconstruction, we expect Σ to be also a correct reconstruction except that with somewhat degraded approximation constants. The computation of Σ is done individually for each triangle $t = \triangle pqr$ in Σ'.

In summary, for each $t \in \Sigma'$, two steps are necessary:

1. Collect the set of relevant points $N_t \subseteq P$.
2. Compute $\mathrm{Del}_{|t}(N_t)$.

In Step 1, we seek a set such that $\mathrm{Del}_{|t}(N_t) = \mathrm{Del}_{|t}(P)$. At the same time, we do not want to take too many points. Let s_t be the circumsphere of t with center in the plane of t and b_t the corresponding ball. A possible solution is to collect those $q \in Q$ whose decimation ball $B(q, \widetilde{R}_q)$ intersect b_t. A simpler procedure which captures some more points is to collect the set Q_t of those $q \in Q$ that are contained in b_t enlarged by a certain constant factor. Here, we use again approximate range reporting. Then $N_t = \bigcup_{q \in Q_t} D_q$, the set of sample points decimated by the sample points in Q_t.

In Step 2, after appropriate geometric transformations, $\mathrm{Del}_{|t}(N_t)$ can be computed using a 3-d convex hull algorithm. So the running time is $O(|N_t| \log |N_t|)$.

Using the local uniformity property of Q, we can verify that $\sum_t |N_t| = O(n)$ and so the overall running time of this step is $O(n \log n)$.

Correctness. We want to verify a correctness guarantee for the reconstruction Σ similar to that for the COCONE algorithm. The main geometric claim in this direction is that the Voronoi edges that intersect Σ' are nearly orthogonal to it. This will imply that the corresponding dual triangles in

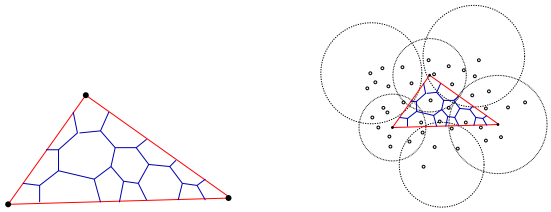

Figure 7: The 3-d Voronoi diagram restricted to a triangle is computed using all the relevant points that were decimated.

Σ have normals close to that of the surface, and that the reconstruction is topologically correct. The proof is as in [2].

The fact that the Voronoi edges that intersect Σ' are nearly orthogonal implies that Σ approximates S pointwise and in normal, and also that Σ has the correct topology (because it implies that $\mathrm{Del}_{|\Sigma'}(P)$ satisfies the ball property [14]).

11 Concluding Remarks

There are several issues that deserve further investigation:

(a) The constants in the running time and storage space need to be improved before the algorithm becomes practical. In particular, WSPDs should be avoided if possible.

(b) We would like to be able to estimate the local feature size efficiently, say in overall time $O(n \log n)$. Then, with our decimation procedure, we could do surface simplification to the extent allowed by a particular ϵ.

(c) We would like to handle surfaces with borders and sharp edges. This is considerably harder. Specifically, the desired algorithm should output a description of the borders and edges detected. Here, there seems to be a connection to the curve reconstruction problem: detecting the border and edges can be seen as a variant of that problem.

(d) The framework used ignores completely the possible anisotropy in the curvature of the surface. The reconstruction of our decimated sampling has triangles that have good aspect ratio. In reality, triangles that are elongated in the direction of smaller curvature might be more desirable.

(e) We also ignore completely additional information that may be available with the capture of the data. In particular, with the viewpoint information.

(f) Finally, we ignore noisy input.

Acknowledgements. The authors acknowledge discussions with Pankaj K. Agarwal, Tamal K. Dey, Jeff Erickson, Sariel Har-Peled, Susan Hert and Kurt Mehlhorn.

References

[1] N. Amenta and M. Bern. Surface reconstruction by Voronoi filtering. *Disc.Comput. Geom.* **22** (1999), 481–504. Earlier version appeared in *Proc. 14th ACM Sympos. Comput. Geom.*, 39-48, 1998.

[2] N. Amenta, S. Choi, T. K. Dey and N. Leekha. A simple algorithm for homeomorphic surface reconstruction. In *Proc. 16th ACM Sympos. Comput. Geom. (SoCG 00),* 213–222, 2000. To appear in *Int. J. Comput. Geom. Appl.*

[3] N. Amenta, S. Choi and R K. Kolluri. The Power Crust, Union of Balls, and the Medial Axis Transform. Submitted to *Int. J. Comput. Geom.*

[4] S. Arya, D. M. Mount, N. S. Netanyahu and R. Silverman. An optimal algorithm for approximate nearest neighbor searching in fixed dimension. *J. ACM* 45(6):891–923, (1998).

[5] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva and G. Taubin. The ball-pivoting algorithm for surface reconstruction. In *IEEE Trans. Visualization Comput. Graphics* **5** (1999), 349–359.

[6] J. D. Boissonat. Geometric structures for three-dimensional shape reconstruction. *ACM Trans. Graphics* **3** (1984) 266–286.

[7] J. D. Boissonat and F. Cazals. Smooth surface reconstruction vial natural neighbor interpolation of distance functions. In *Proc. 16th ACM Sympos. Comput. Geom. (SoCG 00),* 223–232, 2000.

[8] P. Callahan and R. Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest neighbors and n-body potential fields. *J. ACM* **42** (1995), 67–90.

[9] T. M. Chan, J. Snoeyink, and C.-K. Yap. Output sensitive construction of polytopes in four dimensions and clipped Voronoi diagrams in three. In *Proc. 6th ACM-SIAM Sympos. Discr. Algorithms (SODA 95)*, 282–291, 1995. Journal version in *Disc. Comput. Geom.*

[10] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proc. ACM Siggraph,* 303–312, 1996.

[11] T. K. Dey, S. Funke and E. A. Ramos. Surface Reconstruction in Almost Linear Time under Locally Uniform Sampling. *European Workshop on Computational Geometry,* Berlin, March 2001.

[12] C. A. Duncan, M. T. Goodrich and S. G. Kobourov. Balanced aspect ratio trees: Combining the advantages of k-d trees and octrees. In *Proc. 9th ACM-SIAM Sympos. Discr. Algorithms (SODA 98),* 1998.

[13] H. Edelsbrunner. *Algorithms in Combinatorial Geometry.* Springer Verlag, Heidelberg, 1987.

[14] H. Edelsbrunner and N. Shah. Triangulating topological spaces. In *Proc. 10th ACM Sympos. Comput. Geom. (SoCG 94),* 285–292, 1994.

[15] J. Erickson. Nice point sets can have nasty Delaunay triangulations. *Proc. 17th ACM Sympos. Comput. Geom. (SoCG 01),* 96–105, 2001.

[16] M. Gopi, S. Krishnan and C. T. Silva. Surface reconstruction based on lower dimensional localized Delaunay triangulation. In *Eurographics 2000*, M. Gross and F. R. A. Hopgood (Guest Eds.) Blackwell Publishers, 2000.

[17] H. Hoppe, T. Derose, T. Duchamp, J. McDonald and W. Stuetzle. Surface reconstruction from unorganized point clouds. In *Proc. of ACM Siggraph,* 71–78, 1992.