Smooth Surface Reconstruction via Natural Neighbour Interpolation of Distance Functions

Jean-Daniel Boissonnat, Frédéric Cazals

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Smooth Surface Reconstruction via Natural Neighbour Interpolation of Distance Functions

Jean-Daniel Boissonnat — Frédéric Cazals

N° 3985

Aout 2000

THÈME 2

ISSN 0249-6399 ISRN INRIA/RR--3985--FR+ENG

Smooth Surface Reconstruction via Natural Neighbour Interpolation of Distance Functions

Jean-Daniel Boissonnat, Frédéric Cazals

Thème 2 — Génie logiciel
et calcul symbolique

Projet Prisme

Rapport de recherche n° 3985 — Aout 2000 — 28 pages

Abstract: We present an algorithm to reconstruct smooth surfaces of arbitrary topology from unorganised sample points and normals. The method uses natural neighbour interpolation, works in any dimension and allows to deal with non uniform samples. The reconstructed surface is a smooth manifold passing through all the sample points. This surface is implicitly represented as the zero-set of some pseudo-distance function. It can be meshed so as to satisfy a user-defined error bound. Experimental results are presented for surfaces in $\mathbb{R}^3$.

Key-words: Computational Geometry, Voronoi diagrams, Natural Neighbor Interpolation, Surface Reconstruction.

This work has been partially supported by the AFIRST program "Factory of the future".

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, B.P. 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : 04 92 38 77 77 - International : +33 4 92 38 77 77 — Fax : 04 92 38 77 65 - International : +33 4 92 38 77 65

Reconstruction de Surfaces Lisses par Interpolation Naturelle de Fonctions Distance

Résumé : On présente une méthode permettant de reconstruire des surfaces lisses de topologies quelconques à partir de points non structurés et des normales à la surface en ces points. La méthode utilise l'interpolation des voisins naturels, s'applique en toutes dimensions et ne suppose pas que l'échantillonnage soit uniforme. La surface reconstruite est une variété lisse interpolant les points et les normales. Cette surface est donnée sous forme implicite comme le passage par zéro d'une pseudo fonction distance. Elle peut être approchée à la précision demandée par une surface polyédrique. Des résultats expérimentaux sont présentés sur des surfaces de $\mathbb{R}^3$.

Mots-clés : Géométrie algorithmique, Diagrammes de Voronoï, Interpolation, Voisins Naturels, Reconstruction de Surfaces.

1 Introduction

We consider the problem of reconstructing surfaces (i.e. $(d-1)$-manifolds of $\mathbb{R}^d$) from a sparse set of unorganised points which are equipped with normal directions. This problem arises in various contexts such as computer graphics, reverse engineering, image processing, mathematics, chemistry and learning theory. Input may come from a variety of sources : laser range scanners, stereo-vision, $3D$ images (such as seismic data or medical images), or mathematical models (such as implicit surfaces). Geometric model compression is another application area where interpolation of surfaces from discrete samples could find applications. The most important case is $d = 3$ but applications can be found for other values of d.

Surface reconstruction has received considerable attention in the past. The main issues are to deal with surfaces of arbitrary topology, to allow non-uniform sampling —featureless areas need fewer samples, and to produce models with provable guarantees, e.g. smooth manifolds that accurately approximate the actual surface. We can distinguish four main approaches.

The *first approach* is inspired by differential geometry. Locally, the surface is considered as the graph of a function. This function is then approximated by triangulating in a moving projection plane [9] or using moving least-square function approximation techniques [22]. Such methods are usually fast but difficult to make robust and general. Especially, handling sparse and non-uniform data sets seems out of reach.

The *second* main approach consists of considering the surface as an elastic membrane. The starting point is a large membrane enclosing the data set. A deformation process is then applied to minimise the energy down to a local minimum. If the initial guess is sufficiently close to the minimiser, these methods are fast and robust against noise. The variational level set formulation has proved to be very effective [26, 30]. However, such methods can only guarantee convergence towards a local minimum that may be different from the true surface.

The *third* main approach is combinatorial. It consists in constructing a geometric data structure such as the Delaunay triangulation of the point set and to extract from this structure a set of facets that approximate the surface. Early results in that direction are the α-shapes of Edelsbrunner et al. [16, 17] and the sculpting method of Boissonnat [9]. Recently, Amenta et al. [1] have proposed a new Voronoi-based surface reconstruction algorithm that performs well in two and three dimensions.

A similar idea has independently been proposed by Melkemi [23]. Since efficient and robust codes are now available to compute Voronoi diagrams and Delaunay triangulations [14], these methods are very fast. Notice however that the algorithm of Amenta et al. requires to add $2n$ so-called poles to the initial sample points and to construct the Voronoi diagram of a set of points that is 3 times as big as the initial data. In two dimensions, theoretical results on the quality of these methods can be found in the work of Attali [4], Bernardini and Bajaj [7]. The algorithm of Amenta et al. [1] is the first one that has provable guarantees in 3 dimensions. These theoretical results hold when the sampling is sufficiently dense. However, these bounds are rarely met in practical applications and, although the algorithm appears to provide reasonable results under less restrictive sampling conditions, the reconstructed surface may not be a manifold, may have additional holes and may not pass through all the sample points when the sample is not dense enough.

The *last* major approach consists in using the input points to define a signed distance function and to compute its zero-set. The surface is therefore regarded as a level surface of an implicit function defined over the entire embedding space. Such methods have been applied to the surface reconstruction problem by Hoppe et al. [21], Bajaj et al. [5, 6], and Curless and Levoy [13]. The algorithm proposed by Hoppe et al. [21] is related to ours. It estimates a tangent plane at the sample points using the k nearest neighbours and uses the distance to the plane as the signed distance function. The zero-set of this function is then sampled at grid points and polygonalized using the marching cube algorithm. The algorithm of Curless and Levoy [13] is similar and tuned to range images. These algorithms require a (at least locally) uniform sampling since otherwise the k nearest neighbours may well be almost collinear, resulting in a poor estimation of the tangent plane. The algorithm of Bajaj et al. [5] computes a distance function from the α-shape of the points. In order to guarantee that the α-shape has the same topology as the actual surface, the sampling must be dense and uniform. No theoretical analysis of these algorithms has been done yet.

Our method combines Voronoi diagrams and implicit functions. It works in any dimension and is suitable for surfaces of arbitrary topology and non-uniform sampling. Its main features are the following.

First, it uses natural neighbour interpolation [28, 29]. Natural neighbours are easily computed from the Voronoi diagram of the sample points. They allow to deal with non uniform samples and to produce smooth interpolations.

Second, the method directly produces a smooth surface without computing an intermediate polyhedral approximation, as it is usually done [5, 20]. The reconstructed surface is a smooth manifold passing through all the sample points, and its normal at such a point is the normal the point is equipped with. The surface is implicitly represented as the zero-set of a signed pseudo-distance function. Such a representation is attractive for computing offsets or boolean operations and for rendering [8]. A polyhedral approximation of the implicit surface can also be provided.

Third, theoretical guarantees can be derived for the quality of the reconstructed surface. Extensive tests in $\mathbb{R}^3$ show that it performs well in practice, even for small sample sets.

Notations

Let $\mathcal{O}$ be a closed subset of $\mathbb{R}^d$ whose boundary $\mathcal{S}$ is a smooth $(d-1)$-manifold. Let $\mathcal{P}$ be a set of n points $p_1, \ldots, p_n$ called *sample points*, on the boundary of $\mathcal{S}$. Each point p_i is equipped with the unit normal $n(p_i)$ to $\mathcal{S}$ at p_i, oriented towards the outside of $\mathcal{O}$. We denote $H(p_i)$ the hyperplane passing through p_i and perpendicular to $n(p_i)$, and $H^+(p_i)$ the half space limited by $H(p_i)$ and opposite to $n(p_i)$. The hyper-plane bisecting two points x and p_i is denoted B_{x,p_i}, while B^+_{x,p_i} refers to the half-space delimited by B_{x,p_i} and containing x.

Paper overview

Section 2 provides the basics of natural interpolation. Its application to surface reconstruction together with the corresponding theoretical guarantees are presented in sections 3 and 4. Sections 5 and 6 are devoted to the algorithmic aspects, implementation issues and reconstructions results.

2 Natural neighbour interpolation

2.1 Natural neighbours

When the data points are non uniformly distributed, defining neighbouring points is an issue. In particular, in order to construct a surface, we would like to be able to link a given point x to sample points that are close to x and at the same time distributed all around x. Simple definitions like points within a certain distance or k nearest points fail in general to insure such a property. Another approach consists in weighting the points depending on their distance to x. However, such weighting functions are not data dependent and therefore also fail in case of sparse data. We prefer to use the so-called natural neighbours introduced by Sibson [28, 29]. We briefly overview some definitions and properties.

The natural neighbours of a point x are defined as the neighbours of x in the Delaunay triangulation of $\mathcal{P} \bigcup \{x\}$. Equivalently, the natural neighbours are the points of $\mathcal{P}$ whose Voronoi cells are chopped off upon insertion of x. More precisely, let V_{p_i} be the Voronoi cell of p_i in the Voronoi diagram of $\mathcal{P}$ and let V_x be the Voronoi cell of x in the Voronoi diagram of $\mathcal{P} \bigcup \{x\}$. The natural region NR_{x,p_i} is the portion of V_{p_i} stolen away by x, i.e. $V_x \bigcap V_{p_i}$. We also have $NR_{x,p_i} = V_{p_i} \cap B^+_{x,p_i}$. See Figure 1.

Let $w_{p_i}(x)$ denote the Lebesgue measure of NR_{x,p_i} —area in dimension two, volume in dimension three. Note that $NR_{x,p_i} = \emptyset$ and $w_{p_i}(x) = 0$ if p_i is not a natural neighbour of x. The natural coordinate associated to p_i is defined by

$$\lambda_{p_i}(x) = \frac{w_{p_i}(x)}{\sum_i w_{p_i}(x)}.$$

The natural coordinates have three important properties [27, 28, 25] :

1. $\lambda_{p_i}(x)$ is a continuous function of x, and is continuously differentiable excepted at the data sites. Moreover,

$$\nabla w_{p_i}(x) = \frac{\sigma_i(X)}{\|XA_i\|} \overrightarrow{XC}_i,$$

where $\sigma_i(X)$ and C_i are respectively the $(d-1)$-volume and the centroid of the Voronoi facet common to $V(X)$ and $V(X, A_i)$.

2. The $\lambda_{p_i}(x)$ are of bounded support if x belongs to the convex hull of $\mathcal{P}$. More precisely, $\lambda_{p_i}(x)$ vanishes outside the union of the balls circumscribing the Delaunay simplices incident to p_i.

3. The $\lambda_{p_i}(x)$ satisfy an identity called the *local coordinate property* (LCP), stating that x is a convex combination of its neighbours :

$$\sum_i \lambda_{p_i}(x)\, p_i = x.$$

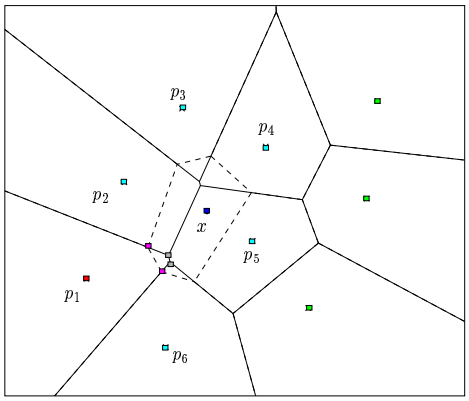

Figure 1: Example of a natural region in $2D$. The natural neighbours of x are $p_1, \ldots, p_6$. The vertices of the natural region NR_{x,p_1} are depicted in grey and black.

When x lies outside the convex hull of $\mathcal{P}$, $w_{p_i}(x)$ is unbounded if p_i is a vertex of the convex hull. In order to keep the $w_{p_i}(x)$ bounded, we need to bound the domain where we want to compute natural coordinates. This can be done quite easily by adding points on a sufficiently large *bounding box*.

An efficient algorithm to compute the natural coordinates in $3D$ is presented in Section 5.

2.2 Natural neighbour interpolation

Assume that to each p_i is attached a continuously differentiable function h_{p_i} from $\mathbb{R}^d$ to $\mathbb{R}$ satisfying $h_{p_i}(p_i) = 0$. We define the natural neighbour interpolation of the h_{p_i} as

$$h(x) = \sum_i \lambda_{p_i}^{1+\omega}(x)\ h_{p_i}(x). \tag{1}$$

for some arbitrarily small $\omega > 0$.

As already pointed out in section 2.1, we use a bounding box $\mathcal{B}$ to bound the natural coordinates of any point inside $\mathcal{B}$. The set of data points consists of the p_i plus some points q_i added on $\mathcal{B}$. We take $h_{q_i} = 0$ for all points q_i on the bounding box. In the sequel, $\mathcal{P}$ will denote the union of the sample points p_i and of the q_i.

For a given x, $h(x)$ is easily evaluated once the $\lambda_{p_i}(x)$ have been computed.

Proposition 1 *h is continuously differentiable and interpolates the h_{p_i}.*

Proof. Since the λ_{p_i} and the h_{p_i} are continuous over $\mathbb{R}^d$, h is continuous over $\mathbb{R}^d$. Moreover, we have $h(p_i) = h_{p_i}(p_i)$ since $\lambda_{p_j}(p_i) = 0$ if $j \neq i$ and $\lambda_{p_i}(p_i) = 1$.

Let us denote by $x^1, \ldots, x^d$ the Cartesian coordinates in $\mathbb{R}^d$. Since the λ_{p_i} are continuously differentiable everywhere except at the p_i, h is continuously differentiable at all $x \notin \mathcal{P}$ and we have

$$\frac{\partial h}{\partial x^j}(x) = \sum_i \lambda_{p_i}^{1+\omega}(x)\ \frac{\partial h_{p_i}}{\partial x^j}(x)\ +\ 1+\omega)\ \sum_i \lambda_{p_i}^{\omega}(x)\ \frac{\partial \lambda_{p_i}}{\partial x^j}(x)\ h_{p_i}(x).$$

When $x \to p_i$, $\frac{\partial h}{\partial x^j}(x) \to \frac{\partial h_{p_i}}{\partial x^j}(p_i)$, which shows that h is continuously differentiable. □

By the implicit function theorem, $h^{-1}(c)$ is continuously differentiable if c is a regular value which, by Sard's theorem is true for almost all c [11, 18].

3 Surface reconstruction

3.1 Interpolation of signed distance functions

In the context of surface reconstruction, various functions can be used for the $h_{p_i}(x)$. We follow Hoppe et al. [21] and use the affine functions $h_{p_i}(x) = (p_i - x) \cdot n(p_i)$, where $n(p_i)$ is the unit normal to $\mathcal{S}$ at p_i. In other words, $h_{p_i}^{-1}(0)$ is the hyperplane tangent to $\mathcal{S}$ at p_i. Other choices should be possible but we do not pursue this issue here.

We then define the interpolating surface $\hat{\mathcal{S}}$ as the zero set $h^{-1}(0)$ and $h^{-1}(\geq 0)$ as the interpolating shape $\hat{\mathcal{O}}$. As mentioned in the previous section, $\hat{\mathcal{S}}$ is almost surely a $(d-1)$-manifold which is continuously differentiable. Moreover, Theorem 6 shows that, when the sampling density increases, the Hausdorff distance between $\hat{\mathcal{S}}$ and $\mathcal{S}$ vanishes.

3.2 Triangulating the reconstructed surface

Given a set of sample points $\mathcal{P}$, the Voronoi diagram of $\mathcal{P}$ uniquely defines the reconstructed surface $\hat{\mathcal{S}}$. However, this surface is implicit and can only be known through the evaluation of h at various points x. We therefore need to compute a discrete approximation of $\hat{\mathcal{S}}$. Since the Delaunay triangulation of $\mathcal{P}$ has to be constructed in order to compute the natural coordinates, we also use it to approximate $\hat{\mathcal{S}}$. The advantages over a standard polygonalization method such as marching cube are twofold. First, the data structure is locally adapted to the geometry of $\hat{\mathcal{S}}$ — instead of uniform. Second, the information of the sample points being on $\hat{\mathcal{S}}$ is fully used —whereas a marching cube recomputes every vertex output.

The dual Voronoi edge of a Delaunay triangle t is the line segment joining the centers of the spheres circumscribing the two tetrahedra sharing t. Define a Voronoi edge as *bipolar* if the implicit function h evaluates to a positive value at an endpoint and to a negative value at its party. Out of the Delaunay triangulation of the sample points, we select the Delaunay facets whose dual edge is bipolar —the *bipolar facets* for short. Corollary 7 states that, under some good sampling condition, the bipolar facets form a polyhedron homeomorphic to $\hat{\mathcal{S}}$. This polyhedron constitutes our initial approximation.

3.3 Refining the initial triangulation

The initial approximation can be refined by inserting points on $\hat{\mathcal{S}}$ and updating the Delaunay triangulation and the set of bipolar facets accordingly. Notice however that in order to perform such a refinement, we cannot use a single Delaunay triangulation since inserting new vertices in it would alter the implicit function. We therefore use two Delaunay triangulations. The first one, built from $\mathcal{P}$, defines the implicit function. The second one, initialised from $\mathcal{P}$, is the triangulation the new points are inserted into. It also provides the Delaunay facets tested for bi-polarity.

More precisely, we adapt Chew's curved-surface meshing algorithm [12] as follows. To each bipolar facet is associated an error, which is the value $h(c)$ at the center of the circle circumscribing the facet. We sort all the bipolar facets by decreasing errors and put all the facets whose errors are larger than a user-specified error bound η in a priority queue Q. We also add to Q the bipolar facets whose angles are not all greater than 30 degrees. While Q is not empty, we extract from Q the facet f with the largest error and search (using binary search) the point s of the Voronoi edge dual to f that lies on $\hat{\mathcal{S}}$ —the so-called surface circumcenter. We then add s to $\mathcal{P}$, update the Delaunay triangulation of $\mathcal{P}$ and the set of bipolar facets. We also update Q by inserting the new bipolar facets whose errors are greater than η.

If the initial set $\mathcal{P}$ satisfies the sampling condition mentioned above, it will remain so when we add points to $\mathcal{P}$. Hence, by Corollary 7, the union of the bipolar facets will remain homeomorphic to $\hat{\mathcal{S}}$ after each insertion of a new point. It follows that the polyhedral approximation $\hat{\mathcal{S}}$ has the same topology as $\hat{\mathcal{S}}$.

Moreover, as shown by Chew, the algorithm halts after a finite number of insertions. When the algorithm terminates, $\hat{\mathcal{S}}$ satisfies the user-specified error bound η.

3.4 Overall reconstruction algorithm

From the previous discussion, performing a reconstruction from a point set $\mathcal{P}$ consists of building the Delaunay triangulation of $\mathcal{P}$ and evaluating the implicit function at the circumcenters of the tetrahedra of that triangulation. The result is a $2D$ triangulation encoding the reconstructed surface. This initial mesh can be refined as explained above.

4 Theoretical guarantees

We first define natural neighbours on a surface. The d-dimensional ball centered on $\mathcal{S}$ that passes through d points of is called a *surface ball.*A *surface facet* is a subset of d points of $\mathcal{P}$ whose corresponding surface ball is *empty*, i.e. that does not contain any sample point in its interior. Clearly, the surface facets are facets of the Delaunay triangulation of $\mathcal{P}$. We define the *surface natural neighbours of* x as the vertices of the surface facets whose corresponding surface balls contain x. It is easily verified that, when $\mathcal{S} = \mathbb{R}^{d-1}$, the surface natural neighbours of a point of $\mathcal{S}$ coincide with the usual natural neighbours in $\mathbb{R}^{d-1}$.

We now recall the definition of a good sampling borrowed from Amenta and Bern [3]. A ball is said to be *maximal* if (1) its interior does not intersect $\mathcal{S}$, (2) it cannot be included in a larger ball satisfying (1). There are two maximal balls passing through a point $x \in \mathcal{S}$.

The *local feature size* $\mathrm{lfs}(x)$ at a point $x \in \mathcal{S}$ is the Euclidean distance from x to the medial axis of $\mathcal{S}$ (i.e. the locus of the centers of the maximal balls). It is easily seen that lfs is Lipschitz, i.e. for any $x, y \in \mathcal{S}$, $\mathrm{lfs}(x) \leq \mathrm{lfs}(y) + \|xy\|$.

$\mathcal{P}$ is called a *ε-sample of* $\mathcal{S}$ if $\mathcal{P} \subset \mathcal{S}$ and if, for all $x \in \mathcal{S}$, there exists a point p_i such that $\|xp_i\| \leq \varepsilon\ \mathrm{lfs}(x)$. When $\varepsilon < 0.1$, the sample is said to be a *perfect* sample.

The following theorem 2 has been proved by Amenta and Bern [10].

Theorem 2 *If $\mathcal{P}$ is a perfect sample, the surface facets form a polyhedron homeomorphic to $\mathcal{S}$.*

As a consequence, when $\mathcal{P}$ is a perfect sample, the Delaunay simplices can be classified as *internal* or *external* depending whether or not they belong to the closed region bounded by the union of the surface facets. The surface facets are common to an internal simplex and to an external simplex and therefore deserve their name.

When the sampling density is large, the Voronoi cells are elongated : their width tends to zero with ε while their diameter tends towards a non-zero quantity that depends on $\mathcal{S}$. Accordingly, a natural coordinate of a point x of $\mathcal{S}$ will be large if the corresponding sample point is a surface natural neighbour of x and small otherwise. This is precisely stated in the following theorem proved in [10].

Theorem 3 *Let x be a point of $\mathcal{S}$ and assume that the centers of the two maximal balls passing through x are not focal points of $\mathcal{S}$. If $\mathcal{P}$ is a perfect ε-sample of $\mathcal{S}$, we have*

$$\sum_{i \notin \mathcal{S}(x)} \lambda_{p_i}(x) = O(\varepsilon), \tag{2}$$

where $\mathcal{S}(x)$ denotes the set of indices of the surface natural neighbours of x.

The next theorem, also proved in [10] states that a subset of the Voronoi vertices, the so-called *poles*, converge towards the medial axis when the sampling density increases. The internal (external) pole of a sample point p_i is the Voronoi vertex of the Voronoi cell of p_i lying inside (outside) $\mathcal{O}$ and at farthest distance from p_i. This result has been independently proved by Amenta and Koluri [2].

Theorem 4 *Let $\mathcal{P}$ be a perfect ε-sample of $\mathcal{S}$ and x a point of $\mathcal{S}$. Assume that the centers of the two maximal balls passing through x are not focal points of $\mathcal{S}$. Then there exists a constant ρ such that, for any sufficiently small ϵ, $\|v_i c_i\| \leq \rho\varepsilon$, where v_i is the pole of p_i and c_i is the center of the maximal ball passing through p_i.*

Theorem 5 *Let $\mathcal{P}$ be a perfect ε-sample of $\mathcal{S}$. For any x on $\mathcal{S}$, we have $|h(x)| = O(\varepsilon)$. For any $\delta > 0$ and for any x that lies at distance at least δ from $\mathcal{S} \cup \mathcal{B}$, there exists an ε and a strictly positive constant η such that $h(x)$ is positive if $x \in \mathcal{O}$, negative otherwise, and $|h(x)| \geq \eta$.*

1. Let $x \in \mathcal{S}$ and $d(x) = \sup_{y \in \mathcal{S} \cup \mathcal{B}} \|xy\|$. We have, using Theorem 3,

$$\begin{aligned} |h(x)| &\leq \sum_i \lambda_{p_i}(x)\, |h_{p_i}(x)| \\ &\leq \sum_{i \in \mathcal{S}(x)} \lambda_{p_i}(x)\, |h_{p_i}(x)| \,+\, d(x)\, O(\varepsilon) \\ &\leq \sum_{i \in \mathcal{S}(x)} \lambda_{p_i}(x)\, \|x p_i\| + d(x)\, O(\varepsilon). \end{aligned}$$

We prove now that, for any surface natural neighbour p_i of x, $\|x p_i\| \leq \frac{2\varepsilon}{1-\varepsilon} \operatorname{lfs}(x)$. Indeed, let S be a surface ball whose interior contains x. Let v be the center of S and

p_i one of the sample points lying on S. We have $\|vx\| \leq \|vp_i\|$ and $\|vp_i\| \leq \varepsilon \operatorname{lfs}(v)$ since $\mathcal{P}$ is a ε-sample. Moreover, $\|xp_i\| \leq \|xv\| + \|vp_i\| \leq 2\varepsilon \operatorname{lfs}(v)$ and $\operatorname{lfs}(v) \leq \operatorname{lfs}(x) + \|xv\| \leq \operatorname{lfs}(x) + \varepsilon \operatorname{lfs}(v)$. Hence $\operatorname{lfs}(v) \leq \frac{1}{1-\varepsilon} \operatorname{lfs}(x)$, which proves the claim.

We therefore have $|h(x)| \leq (\frac{2\varepsilon}{1-\varepsilon} \operatorname{lfs}(x) + d(x))\ O(\varepsilon)$. Since both $\operatorname{lfs}(x)$ and $d(x)$ are bounded quantities that do not depend on ε, we have proved the first part of the theorem.

2. Let $x \in \mathcal{S}$ and assume, as in the theorem, that x lies at distance at least δ from $\mathcal{S} \cup \mathcal{B}$ for some constant $\delta > 0$. We therefore have $\|xp_i\| \geq \delta$ for any $p_i \in \mathcal{P}$. Without loss of generality, we assume that $x \in \mathcal{O}$, the other case being entirely symmetrical.

Consider first the case of an infinitely dense sampling so that any point on $\mathcal{S}$ is a sample point. In such a case, the set $\mathcal{N}(x)$ of all natural neighbours of x is a portion of $\mathcal{S}$. This set consists of the points of $\mathcal{S}$ whose maximal balls contain x (see Figure 2).

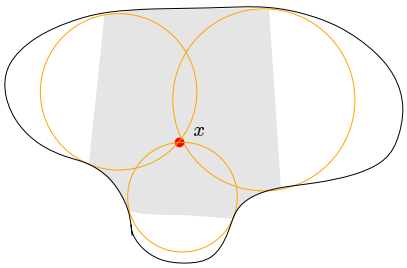

Figure 2: The set $\mathcal{N}(x)$.

Plainly, any point $p \in \mathcal{N}(x)$ is visible from x, i.e. the relative interior of the line segment $]xp[$ does not intersect $\mathcal{S}$. It follows that $h_p(x) \geq 0$ for any $p \in \mathcal{N}(x)$. Moreover, $h_p(x) \geq \eta > 0$. Indeed, let p be a natural neighbour of x. The maximal ball tangent to $\mathcal{S}$ at p contains x. Let R denotes its radius and let α be the angle between the plane tangent to $\mathcal{S}$ at p and the vector $\overrightarrow{px}$ (see Figure 3). We have $h_p(x) = \|xp\| \sin\alpha$ and $2R \sin\alpha \geq \|xp\|$. Hence, if R_{max} denotes the maximum radius of all the empty spheres, we have $h_p(x) \geq \frac{\delta^2}{2R_{max}} \stackrel{\text{def}}{=} \eta$. Since R_{max} is bounded, $\eta > 0$. It follows that

$$h(x) = \int_{\mathcal{N}(x)} \lambda_p(x)\, h_p(x)\, ds \geq \eta > 0.$$

This proves the theorem in the limit.

Let us now consider the case of a finite sample. The above argument has shown that $\mathcal{N}(x)$ is strictly included inside $\mathcal{V}(x)$, the set of points of $\mathcal{S}$ that are visible from x. Therefore, $h_{p_i} \geq \eta > 0$ for all $p_i \in \mathcal{N}(x)$. Let p_i be a natural neighbour of x not in $\mathcal{N}(x)$. (Remind that $\mathcal{N}(x)$ is the portion of $\mathcal{S}$ consisting of all the natural neighbours of x in the continuous setting.) We aim at proving that, for a sufficiently small ε, $p_i \in \mathcal{V}(x)$.

Since p_i is a natural neighbour of x, there exists at least one Delaunay simplex incident to p_i whose circumscribing ball contains x. Assume first that x belongs to the Delaunay ball D_{p_i} centered at the pole of p_i. Since $p_i \notin \mathcal{N}(x)$, the maximal ball B_{p_i} that passes through p_i does not contain x. But, since by Theorem 4, D_{p_i} and B_{p_i} are close and tends to become identical when ε tends to zero, there must exist a point z close to x (and converging towards x when ε vanishes) such that $p_i \in \mathcal{N}(z)$. By continuity of the natural coordinates and the fact that $\mathcal{N}(x)$ is strictly included inside $\mathcal{V}(x)$, it follows that, for any sufficiently small ε, p_i belongs to $\mathcal{V}(x)$.

Assume now that x does not belong to D_{p_i} but to some other Delaunay ball D passing through p_i. We first observe that the radius of D is at least $\frac{\delta}{2}$ since $\|xp_i\| \geq \delta$. Moreover, the Voronoi cell of p_i is elongated : its width is $O(\varepsilon)$ while its diameter is $O(1)$. Easy computations then show that D is included in the Minkowski sum of D_{p_i} and a ball of radius $O(\varepsilon)$. The rest of the proof is the same as for the previous case. □

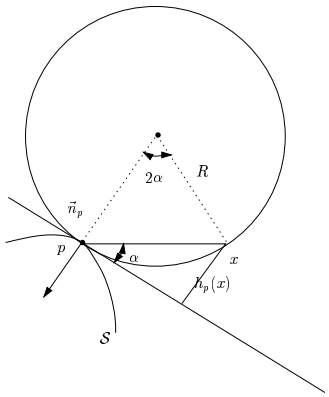

Figure 3: Bounding $h_p(x)$.

Corollary 6 *Let $\mathcal{P}$ be a perfect ε-sample of $\mathcal{S}$. The Hausdorff distance between $\mathcal{S}$ and $\hat{\mathcal{S}}$ tends to zero when ε tends to zero.*

Proof. We first show that, for any $x \in \mathcal{S}$, there exists $y \in \hat{\mathcal{S}}$ such that $\|xy\| = O(\varepsilon)$. Since $\mathcal{P}$ is a perfect ε-sample, for any $x \in \mathcal{S}$, there exists $p_i \in \mathcal{P}$ such that $\|xp_i\| \leq \varepsilon \operatorname{lfs}(x)$, which proves the claim since p_i belongs also to $\hat{\mathcal{S}}$.

Conversely, let $y \in \hat{\mathcal{S}}$. By theorem 5, $\inf_{x \in \mathcal{S}} \|xy\|$ tends to zero when ε tends to zero. □

Corollary 7 *If $\mathcal{P}$ is a ε-sample of $\mathcal{S}$ for some sufficiently small ε, the bipolar facets form a polyhedron homeomorphic to $\hat{\mathcal{S}}$.*

Proof. Let $y \in \hat{\mathcal{S}}$. By the above corollary, for any $\eta > 0$ and a sufficiently small $\varepsilon > 0$, there exists $x \in \mathcal{S}$ such that $\|yx\| \leq \eta$. Since $\mathcal{P}$ is a perfect ε-sample, there exists $p_i \in \mathcal{P}$ such that $\|xp_i\| \leq \varepsilon \operatorname{lfs}(x)$. We therefore have $\|yp_i\| \leq \|yx\| + \|xp_i\| = \eta + O(\varepsilon)$. Let $\hat{\operatorname{lfs}}$ be the local feature size on $\hat{\mathcal{S}}$. $\hat{\operatorname{lfs}}$ is bounded from below since $\hat{\mathcal{S}}$ is continuously differentiable. It follows that, for a sufficiently small ε, $\mathcal{P}$ is a perfect sample of $\hat{\mathcal{S}}$ and Theorem 2 applied to $\hat{\mathcal{S}}$ allows to conclude. □

5 Efficiently computing the natural coordinates in $3D$

5.1 Volume computation algorithm

5.1.1 Preamble

Consider the natural region NR_{x,p_i} associated to p_i upon insertion of x. Since this region is delimited by Voronoi centers in convex position, one could just collect them, compute the convex hull and its volume. But this requires a convex hull algorithm. Another method proposed in [24] consists of first computing the Voronoi region of x, and second subdividing it into the natural neighbours sub-regions. This strategy requires intersection algorithms.

To keep up with the Delaunay machinery, we calculate *simultaneously* the volume of all the natural regions by mimicking the insertion of x in the Delaunay triangulation.

Before describing the algorithm more precisely, we recall how a point is inserted in a Delaunay triangulation, and describe the geometry of a natural region.

5.1.2 Inserting a point in a $3D$ Delaunay triangulation

Inserting a point x in a Delaunay triangulation is a two stages process. First, the tetrahedra in conflict i.e. whose circumscribing ball contains x are sought. The union of these tetrahedra forms a star-shaped region referred to as the *cavity*. Second, this cavity is filled up with new tetrahedra. Each such tetrahedron is defined by a Delaunay facet on the boundary of the cavity and the new vertex inserted. See Figure 4 for a $2D$ example.

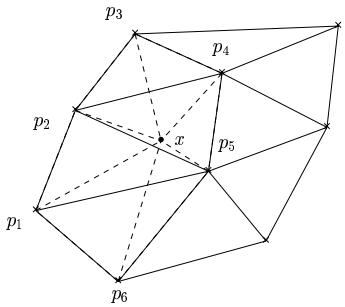

Figure 4: Inserting a point in a $2D$ Delaunay triangulation

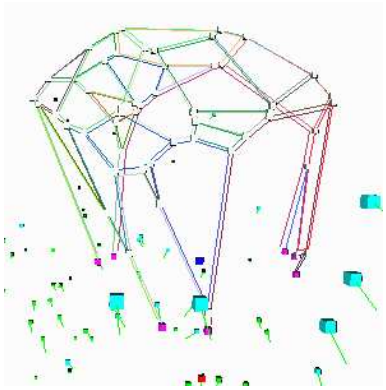

Figure 5: Example natural region in $3D$

5.1.3 Geometry of a natural region

As already mentioned, NR_{x,p_i} consists of V_{p_i} chopped off by the half-space B^+_{x,p_i}. If a facet f of V_{p_i} verifies $f \cap B_{x,p_i} = \emptyset$, call it a *cap*. If on the opposite $f \cap B_{x,p_i} \neq \emptyset$, call the intersection $f \cap B^+_{x,p_i}$ an *arch*. The region NR_{x,p_i} consists of caps and arches, together with the bottom facet defined by $V_{p_i} \cap B_{x,p_i}$. See Figure 5 for an illustration.

Suppose now that a Voronoi facet —arch or cap— is available as the ordered sequence of its vertices. Since the facet is convex, this sequence provides a natural triangulation: the fan-shaped triangulation radiating out any vertex. The union of these triangles provides a triangulation $\mathcal{T}_i$ of all the facets of NR_{x,p_i} but the bottom one.

Once such a triangulation is available, if o_i is any vertex of the bottom facet, the volume of NR_{x,p_i} reduces to the sum of the volumes of the tetrahedra defined by o_i and the triangles of $\mathcal{T}_i$.

5.1.4 The algorithm

From the previous discussion, computing the volume reduces to reporting the sequences of vertices defining the caps and the arches. To see how this can be done, define an external (internal) edge of the cavity as a Delaunay edge located on the boundary of (inside) the cavity.

Let e be an external edge. Upon the insertion of x, some tetrahedra $\{t_1, \ldots, t_k\}$ adjacent to e are killed and replaced by two new tetrahedra $\{t_1^+, t_2^+\}$. These new tetrahedra have x as apex and the Delaunay facets adjacent to e as base. Let $\{cc_i^+\}_{i=1,2}$ and $\{cc_i\}_{i=1,\ldots,k}$ be the circumcenters of the new and old tetrahedra. The sequence $\{cc_1^+, cc_1, \ldots, cc_k, cc_2^+\}$ describes the arch intersection of the dual facet of e with B^+_{x,p_i}.

Consider now an internal edge (p_i, p_j). Such an edge does not exist in the triangulation resulting from the insertion of x. But its dual facet is common to NR_{x,p_i} and NR_{x,p_j} and is a cap.

To summarise, the volume computation is a two stages process. First, the tetrahedra in conflict are sought. Second, the edges of the cavity are rotated around. These rotations report all the facets of all the natural regions but the bottom facets. If $e = (p_i, p_j)$ is a cavity edge, its dual is a cap or an arch which is used to increment the volume of p_i and p_j. The complexity of the volume computation is the same as the one of the Delaunay insertion.

5.2 Experimental observations

5.2.1 Overhead of the natural coordinates computation

As explained above, computing the natural coordinates of a point consists of simulating its insertion in a Delaunay triangulation. However, with reference to a straight Delaunay insertion, this requires maintaining auxiliary data structures, computing new circumcenters, as well as computing volumes.

To measure this overhead, we ran two programs for various models: one performing a sequence of *insert()* operations, and one executing a sequence of *simulateIn-*

sert()+insert() operations. Notice that this test bed ensures that the *simulateInsert()* and *insert()* operations are performed on triangulations of the same size. The results are listed in Table 1 whose five columns respectively report the random model, the sample size, the insertion time t_1, the cumulated time $t_1 + t_2$ of the insertions simulations followed by the insertions, and the ratio t_2/t_1. Several observations can be raised.

Localising the tetrahedra in conflict against re-triangulating the cavity. Computing the natural coordinates incurs a penalty factor ranging from 10 to 5 depending on the number of points processed. This variation corresponds to the cost of finding the tetrahedra in conflict versus re-triangulating the cavity. The larger the sample, the more prominent the relative localisation cost.

Overhead of the natural coordinates computations. For a given sample size, the overhead is almost constant, which corresponds to the extra operations detailed in section 5.

model	#points	insertion (t_1)	insertion (t_1) + simulation (t_2)	t_2/t_1
inCube	1000	0.09	0.94	9.44
onCube	1000	0.13	1.03	6.92
inSphere	1000	0.09	0.89	8.89
onSphere	1000	0.07	0.67	8.57
inCube	5000	0.58	4.85	7.36
onCube	5000	0.74	6.26	7.46
inSphere	5000	0.59	4.88	7.27
onSphere	5000	0.43	3.38	6.86
inCube	10000	1.31	9.99	6.63
onCube	10000	1.70	13.49	6.94
inSphere	10000	1.32	10.08	6.64
onSphere	10000	0.99	7.01	6.08
inCube	50000	7.42	52.27	6.04
onCube	50000	10.21	75.06	6.35
inSphere	50000	7.47	52.54	6.03
onSphere	50000	5.63	36.32	5.45
inCube	100000	15.52	105.91	5.82
onCube	100000	21.00	149.28	6.11
inSphere	100000	15.54	106.48	5.85
onSphere	100000	11.73	73.61	5.28

Table 1: Computing natural coordinates versus inserting in Delaunay

5.2.2 Profiling the natural coordinates computation

A typical profile over a sequence of *simulateInsert()* calls is summarised by Table 2. Apart from the system related procedures which are up to 22% of the computational time, the top 15 most offending functions contribute to 60% of the cost.

Figuring out the tetrahedra in conflict with the point tested corresponds to f_6. Functions f_{10}, $f_4$and f_{11} report the cost of rotating around the external and internal edges of the cavity. The costs of computing the circumcenters of the tetrahedra created during the simulation is accounted for by f_3 and f_{12}. Functions f_1 and f_7 are the backbone of the volume computation. In particular f_7 takes as input two indices in stack of circumcenters and processes the corresponding Voronoi facet or arch. Elementary volume computations are subsequently dispatched in f_2, f_8, f_9, f_{13} and f_{14}. This even distribution of costs shows that a totally different technique may be necessary to reduce the 5 to 10 overhead just mentioned.

%time	function name
11.3	f_1 computeVolume(array1<point3d*>&, int, int,point3d*, int&)
9.1	f_2 operator -(const point3d&, const point3d&)
5.0	f_3 Tetra::sphereParams(point3d&, point3d&, point3d&, point3d&, point3d&, double&)
4.7	f_4 NRDelaunay::simulateCreate(Tetra*, int)
3.5	f_5 array1<point3d*>::operator [](long) const
2.8	f_6 AR_TOOLS<Vertex, Tetra>::Insphere(Tetra*, Vertex&)
2.7	f_7 faceContributionToNRVolume(Vertex*, int, int, int, float)
2.7	f_8 vector::vector(double, double, double)
2.7	f_9 operator *(vector&, vector&)
2.5	f_{10} Tetra::cw_index(Vertex*, Vertex*, int&, int&) const
2.4	f_{11} NRDelaunay::internalEdges_new(void)
2.2	f_{12} Tetra::getMyCC(void)
2.1	f_{13} operator *(vector&, vector&)
2.1	f_{14} mixed(vector&, vector&, vector&)
2.0	f_{15} Tetra::resetFlags(void)

Table 2: Profile over a sequence of *simulateInsert()*

5.3 About the Delaunay triangulation used

Practically, we use the randomised Delaunay triangulation algorithm of Devillers [14]. This code processes about 500,000 points randomly distributed in the unit cube per minute on a Pentium-III at 500MHz processor with 256MB of RAM. Notice however that when the points are not distributed in a volume but on a surface, which is the case in the reconstruction context, the performances are slightly worse since the connectivity of the triangulation is higher.

6 Reconstruction results

In addition to a couple of synthetic models —the sphere and the double torus depicted on Figures 6 and 7, we ran the reconstruction algorithm on several models: 4 bunnies

ranging from 250 to 8141 points, a triceratops, a cow, a horse, and a molecular surface.

Practically, the value of ω in the expression of $h(x)$ is set to 0 —see Equation 1 of Section 2.2.

As explained in section 2.1, the model sample points are added a set of points located on a bounding box. This box is chosen as the model tight bounding box slightly expanded. We explored the range $[1.2, 2]$ by steps of 0.1 for the expansion coefficient, and the range $i \times i, i = [2, 8]$ for the number of points per bounding box face. The surface reconstructed is not sensitive to these factors in the sense that one gets the same bipolar facets.

The surface polygonalization returned is the initial set of bipolar Delaunay facets for the big bunny, the cow, the horse and the molecule. For the other models, this mesh is refined down to $\eta = 1/3$ of the initial maximum grade η_0.

The method performs well on all models. Good results are produced even for very small samples. In Figures 12 and 13, results are presented for a set of only 250 points from the bunny. It is to be observed that refining the mesh leads to a neat improvement on the ears where the sampling is very crude and does not satisfy the good sampling condition. For this kind of data and to the best of our knowledge, no combinatorial method produce a nice polyhedral surface using only the data points as vertices.

The same observations can be done on the cow and the triceratops models. The method provides satisfactory results even on the thin parts like the horns. Figure 15 shows again that refining the mesh is quite effective and allows to remove some singularities that occur if the mesh is not refined.

For each model, Table 3 lists the following parameters: the number of sample points, the number of tetrahedra in the triangulation, the number of facets reported by the triangulation algorithm, the average and maximum number of natural neighbours over the insertion simulations, the time required by the meshing algorithm —in seconds, and the number of evaluations of the pseudo-distance function h that are performed per minute.

The main observation is that the running times may vary a lot for a given number of sample points. In particular, the bunny with 8k points takes longer that any other

model. Not surprisingly, the number of natural neighbours that are processed at each evaluation of the function is much higher.

Finally, let us note that the algorithm is numerically quite robust. The Delaunay triangulation is computed exactly using an exact evaluation of the geometric predicates, most notably the `Insphere` test that decides if a point p lies inside, on, or outside the sphere passing through four other points q, r, s and t. The algorithm we use represents the input coordinates as 24-bits integers and uses a very efficient static filter [14]. The computation of the natural coordinates is less sensitive to round-off errors and uses floating point arithmetic.

model	#points	#tetra.	#facets	average #NNs	max #NNs	time
bunny-1000-	999	7474	2000	55.29	228	14.60
bunny2	8141	55079	16276	185.64	1014	526.40
cow	3069	21108	6202	48.50	190	40.14
horse	11136	74080	22006	88.04	453	292.15
papaine2	18072	132749	36130	68.75	227	398.68
bunny-250- η=.01	246	2670	485	29.31	88	1.63
bunny-250- η=.003	246	2569	853	22.96	88	4.50
bunny-500- η=.006	489	4159	976	43.41	159	5.37
bunny-500- η=.002	489	4260	1200	33.28	159	6.96
triceratops η=.15	2507	17246	5016	46.75	155	31.18
triceratops η=.05	2507	17360	5652	41.21	157	41.62

Table 3: Statistics

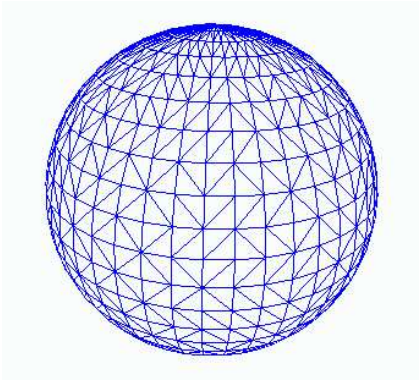

Figure 6: Sphere reconstruction —928 points

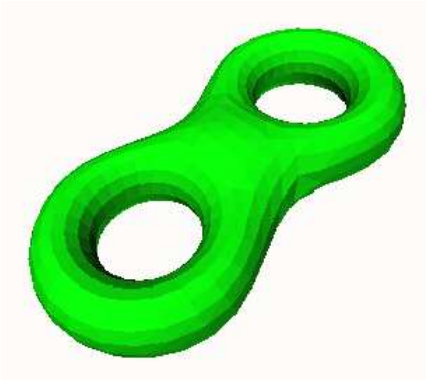

Figure 7: Figure eight reconstruction —768 points

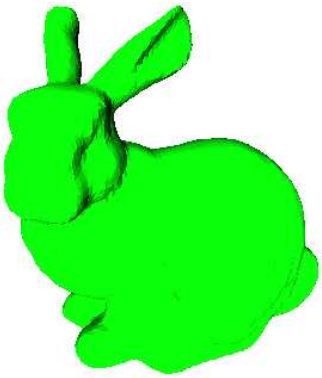

Figure 8: Bunny2 reconstruction

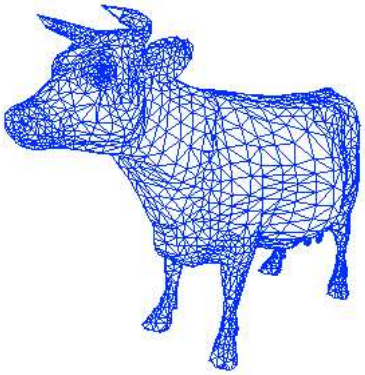

Figure 9: Cow reconstruction

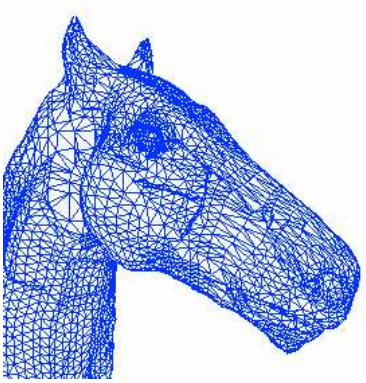

Figure 10: Horse reconstruction, zoom on the head

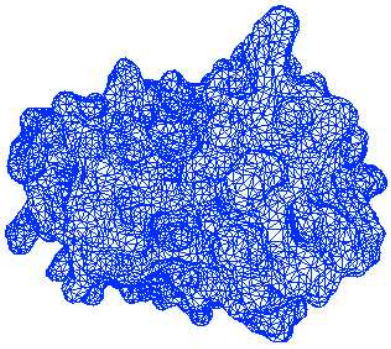

Figure 11: Papaine reconstruction

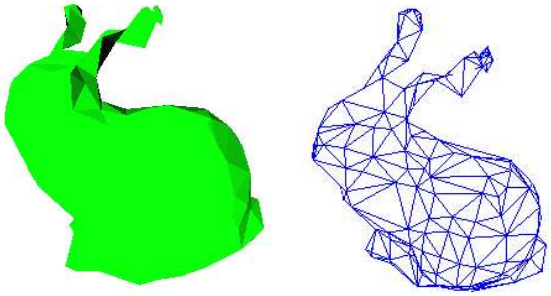

Figure 12: Bunny-250 reconstruction

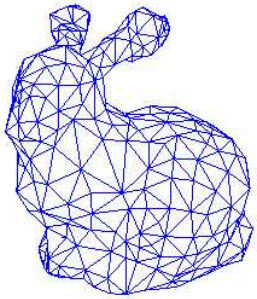

Figure 13: Bunny-250 reconstruction, with refined mesh

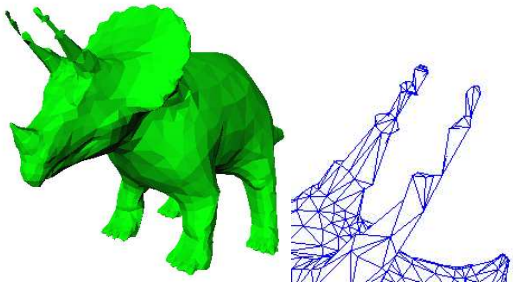

Figure 14: Triceratops reconstruction

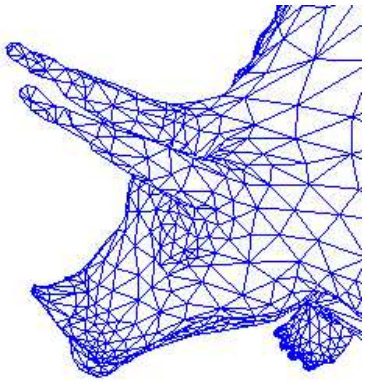

Figure 15: Triceratops reconstruction, refined mesh

7 Conclusion

Our method is based on two main assumptions. First, we have assumed that the normals to the surface at the sample points are known. Although, normals are provided in some applications, in some others, they need to be estimated from the points. The normals can be estimated by approximating locally the tangent plane or can be deduced from the Voronoi diagram of the sample points, as suggested by Amenta and Bern [3].

We have also assumed that the surface is smooth and without boundary. We are currently extending this approach to surfaces with sharp edges and surfaces with boundaries. Results will be reported in forthcoming reports.

We have only considered surfaces, i.e. manifolds of codimension 1. It should be interesting to investigate other cases, most notably the case of curves in $\mathbb{R}^3$. For previous work on this problem, see [15, 19, 30].

Acknowledgements

Stéphane Nullans is acknowledged for preliminary contributions to this problem. Frank Da and Mariette Yvinec deserve special thanks for frequent discussions. Last but not the least, we express gratitude to Olivier Devillers for his $3D$ Delaunay hierarchy.

References

[1] N. Amenta, M. Bern, and M. Kamvysselis. A new Voronoi-based surface reconstruction algorithm. In *Proc. SIGGRAPH '98*, Computer Graphics Proceedings, Annual Conference Series, pages 415–412, July 1998.

[2] N. Amenta and R. Kolluri. Accurate and efficient unions of balls. In *ACM Symposium on Computational Geometry*, 2000.

[3] Nina Amenta and Marshall Bern. Surface reconstruction by Voronoi filtering. *Discrete Comput. Geom.*, 22(4):481–504, 1999.

[4] D. Attali. r-regular shape reconstruction from unorganized points. *Comput. Geom. Theory Appl.*, 10:239–247, 1998.

[5] C. Bajaj, F. Bernardini, and G. Xu. Automatic reconstruction of surface and scalar fields from 3d scans. *Comput. Graph.*, ??:109–118, 1995. Proc. SIGGRAPH'95.

[6] F. Bernardini, C. Bajaj, J. Chen, and D. Schikore. Triangulation-based object reconstruction methods. In *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pages 481–484, 1997.

[7] Fausto Bernardini and Chandrajit L. Bajaj. Sampling and reconstructing manifolds using alpha–shapes. In *Proc. 9th Canad. Conf. Comput. Geom.*, pages 193–198, 1997.

[8] Jules Bloomenthal, editor. *Introduction to Implicit Surfaces*, volume 391. Morgan-Kaufmann, 1997.

[9] Jean-Daniel Boissonnat. Geometric structures for three-dimensional shape representation. *ACM Trans. Graph.*, 3(4):266–286, 1984.

[10] Jean-Daniel Boissonnat and Frédéric Cazals. Natural coordinates on a surface. Rapport de recherche, INRIA, 2000. to appear.

[11] J. W. Bruce and P. J. Giblin. *Curves and Singularities (2nd edition)*. Cambridge University Press, New York, 1992.

[12] L. P. Chew. Guaranteed-quality mesh generation for curved surfaces. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 274–280, 1993.

[13] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proc. SIGGRAPH 96*, pages 303–312, 1996.

[14] Olivier Devillers. Improved incremental randomized Delaunay triangulation. In *Proc. 14th Annu. ACM Sympos. Comput. Geom.*, pages 106–115, 1998.

[15] T. K. Dey, K. Mehlhorn, and E. R. Ramos. Curve reconstruction : Connecting dots with good reason. In *Proc. 15th Annu. ACM Sympos. Comput. Geom.*, pages 197–206, 1999.

[16] H. Edelsbrunner, D. G. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Trans. Inform. Theory*, IT-29:551–559, 1983.

[17] H. Edelsbrunner and E. P. Mücke. Three-dimensional alpha shapes. *ACM Trans. Graph.*, 13(1):43–72, January 1994.

[18] A.T. Fomenko and T.L. Kunii. *Topological Modeling for Visualization*. Springer, 1997.

[19] J. Giesen. Curve reconstruction, the traveling salesman problem and menger's theorem on length. In *Proc. 15th Annu. ACM Sympos. Comput. Geom.*, pages 207–216, 1999.

[20] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction. In *Proc. SIGGRAPH 94*, pages 295–302, 1994.

[21] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *Comput. Graphics*, 26(2):71–78, 1992. Proc. SIGGRAPH '92.

[22] David Levin. Mesh-independent surface interpolation. In *4th International Conference on Curves and Surfaces*, page 46, 1999. Saint-Malo, France.

[23] M. Melkemi. A-shapes and their derivatives. In *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pages 367–369, 1997.

[24] S. J. Owen. An implementation of natural neighbor interpolation in three dimensions. Master's thesis, ??, 1992.

[25] B. Piper. Properties of local coordinates based on dirichlet tessalations. *Computing*, 8:227–239, 1993.

[26] J. A. Sethian. *Level Set Methods.* Cambridge University Press, 1996.

[27] R. Sibson. A vector identity for the Dirichlet tesselation. *Math. Proc. Camb. Phil. Soc.*, 87:151–155, 1980.

[28] R. Sibson. A brief description of natural neighbour interpolation. In Vic Barnet, editor, *Interpreting Multivariate Data*, pages 21–36. John Wiley & Sons, Chichester, 1981.

[29] David F. Watson. *Contouring: A Guide to the Analysis and Display of Spatial Data.* Pergamon, 1992.

[30] H-K. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit, nonparametric shape reconstruction from unorganized points using a variational level set method. Cam report 98-7, UCLA, http://www.math.ucla.edu/applied/cam/index.html, 1998.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - B.P. 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Lorraine : Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 Villers lès Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot St Martin (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, B.P. 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399