Available online at www.sciencedirect.com
ScienceDirect
Computational Geometry 36 (2007) 106–116
Computational Geometry
Theory and Applications
www.elsevier.com/locate/comgeo

An incremental algorithm for reconstruction of surfaces of arbitrary codimension

Daniel Freedman

Rensselaer Polytechnic Institute, Department of Computer Science, Troy, NY 12180, USA

Received 29 December 2003; received in revised form 29 March 2006; accepted 8 May 2006

Available online 30 June 2006

Communicated by J.-R. Sack

Abstract

A new algorithm is presented for surface reconstruction from unorganized points. Unlike many previous algorithms, this algorithm does not select a subcomplex of the Delaunay Triangulation of the points. Instead, it uses an incremental algorithm, adding one simplex of the surface at a time. As a result, the algorithm does not require the surface's embedding space to be $\mathbb{R}^3$; the dimension of the embedding space may vary arbitrarily without substantially affecting the complexity of the algorithm. One result of using this incremental algorithmic technique is that very little can be proven about the reconstruction; nonetheless, it is interesting from an experimental viewpoint, as it allows for a wider variety of surfaces to be reconstructed. In particular, the class of non-orientable surfaces, such as the Klein Bottle, may be reconstructed. Results are shown for surfaces of varying genus.

Keywords: Surface reconstruction; Point clouds; Incremental algorithm

1. Introduction

This paper treats the problem of surface reconstruction from unorganized points. Given a series of points sampled from a surface (which is unknown), the goal is to construct a second surface which interpolates the points and is sufficiently like the original. Ideally, this means that the second surface should be homeomorphic to the first. Furthermore, it should be geometrically close to the first; for example, the symmetric Hausdorff distance between the two surfaces ought to small. The main difficulty in designing algorithms to meet these specifications arises precisely from the fact that the points are unorganized. This implies that neighborhood information (i.e. topological information) is not immediately obvious, and must be discovered from the samples.

A critical distinction from previous surface reconstruction algorithms is that the current method does not require the surface to be embedded in $\mathbb{R}^3$. Rather, the surface may be embedded in an arbitrary Hilbert space. As a result, we may argue that the current algorithm is more "topologically intrinsic", as it depends only on the manifold itself, and not the space in which the manifold lives. Practically speaking, the algorithm can reconstruct many surfaces which cannot be reconstructed using earlier algorithms; the most interesting of these are perhaps the non-orientable surfaces, such as the Klein Bottle and the real projective plane. The cost of this generalization is the inability to rigorously establish

E-mail address: freedman@cs.rpi.edu (D. Freedman).

0925-7721/$ – see front matter
doi:10.1016/j.comgeo.2006.05.004

claims about the algorithm. At present, we cannot prove any important results about the validity of the algorithm, and indeed, it may be impossible to do so. Instead, we present the algorithm as one which performs well in an experimental context.

The algorithm is able to function in a high-dimensional embedding space because it is *incremental*. At each iteration, the surface is grown by adding a single triangle. A major issue related to such an incremental algorithm is the following: how can we can avoid choosing triangles that lead to surface self-intersections? We deal with this issue by performing all computations *locally*, in a plane which approximates nearby tangent spaces of the manifold. Again, note that the emphasis of this paper is largely *experimental*; no proofs are offered, and the algorithms are motivated from the point of view of geometric and topological intuition. The proof, in this case, is in the pudding, and the results show that the algorithm works well in practice.

It is worth pointing out the relationship between this work and our previous work [1] on manifold reconstruction. That work also, in some sense, involved an incremental algorithm: d-dimensional discs were estimated at each point (d is the dimension of the manifold), and these were stitched together to form a manifold. However, the algorithms are quite different from one another. The previous work does not attempt to compute new patches based on patches that have already been computed; instead, all of the patches are computed simultaneously, and the stitching only occurs afterwards. Thus, this algorithm is not truly incremental. The new algorithm, by contrast, is incremental: in computing a new triangle, we explicitly account for the triangles that have already been computed. (Of course, there is a difference also in the fact that [1] computes patches, i.e. collection of simplices, whereas the new algorithm computes triangles.) This distinction results in a crucial difference in performance. On several of the examples in Section 4, [1] generates surfaces which contain many spurious triangles, and are not true manifolds.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 presents the new algorithm, including a discussion of complexity. Section 4 shows results on a variety of different surfaces, including a couple of examples of non-orientable surfaces.

2. Related work

Surface reconstruction, as a topic of research, is relatively new. One of the earliest papers in the field was that of Boissonnat [2], which presented a method of "sculpting" Delaunay Triangulations. The algorithms of Hoppe et al. [3,4] constructed a function whose zero level-set was the surface. Curless and Levoy [5] made improvements to this algorithm by taking advantage of the type of data that is produced by laser range scanners. Bernardini and Bajaj [6] offered an algorithm, based on α-shapes, with provable properties. However, like much work based on α-shapes, and like later work by Bernardini et al. [7], this algorithm requires relative uniformity of sampling, which is a drawback.

The crust of Amenta and Bern [8] provided the first algorithm which both allowed for highly non-uniform sampling and which provided theoretical guarantees. This algorithm was simplified in [9], as was the proof of homeomorphism. Finally, certain guarantees of the "watertightness" of a surface are given by the power-crust algorithm [10].

Dey and collaborators have used a related notion, the "co-cone", in order to develop several new algorithms. These extensions include: algorithms with better complexity [11,12]; algorithms which are implemented using data structures designed to handle large amounts of data [13]; and algorithms which allow for the detection of undersampling [14] and manifold boundaries [15]. A more recent paper [16] uses the co-cone idea in order to determine the topological dimension of various objects, and presents a reconstruction scheme based on the computed dimension.

The idea of using an incremental approach to reconstruction was first proposed in [2]. Since then, there have been several pieces of work along this line. The Ball-Pivoting Algorithm of Bernardini et al. [7], in which a ball of fixed radius rolls around to find new triangles, is very fast in practice, and has been quite effective in cultural heritage applications. One drawback is its reliance on relatively uniform sampling. Gopi et al. [17,18] uses computed normals for an incremental construction of the surface; these algorithms are also quite fast. Cohen-Steiner and Da [19] use a Delaunay-based scheme, which tries to add triangles in a sensible way, so as to avoid topological pathologies. One should note, as well, the work of Crossno and Angel [20] and Mencl and Muller [21].

Other approaches include that of Adamy et al. [22], who present an algorithm for constructing the surface in an incremental fashion based on the idea of λ-intervals. Boissonat and Cazals [23] provide an algorithm with provable properties based on the concept of nearest neighbor interpolation using the Sibson interpolant. Finally, Edelsbrunner [24] presents a novel algorithm based on ideas from Morse Theory.

3. Incremental reconstruction

In this section, we describe the incremental algorithm for surface reconstruction, which may be applied to any surface without boundary which is embedded in a Hilbert Space of dimension D. The algorithm is given in pseudocode in Fig. 1; smaller algorithms, which are called by the main algorithm, are specified in Figs. 3, 4, 6, and 7. The reader may find it enlightening, before examining these various bits of pseudocode, to read the explanation in the following sections. The explanation will use some of the notation from the latter figures.

The algorithm is incremental in nature: it grows the surface by adding one simplex at a time. At each iteration, the algorithm chooses a "dangling edge" e from the current simplicial complex K: an edge which has only a single coface, f_1. Since the surface is assumed to be without boundary, every edge must have exactly two cofaces; the goal is then to find a second coface f_2 for this dangling edge. This is the manner in which the algorithm is incremental, and is illustrated in Fig. 2. When the new simplex f_2 is added, the dangling edge will no longer be dangling; however, the new simplex may both create new dangling edges, as well as destroy other existing dangling edges. The algorithm terminates when there are no more dangling edges; at this point, we have produced a 2-manifold without boundary.

The main algorithmic question is then: for a given dangling edge e, how does one choose a second coface f_2? We sketch the procedure here, and then elaborate upon these ideas what follows. The algorithm has three main stages:

- *Filtering*: Before f_2 is added, we have a simplicial complex K. We would like to ensure that after adding f_2, we still maintain a simplicial complex; that is f_2 can only intersect the other simplices of K is the "natural" ways allowed for a simplicial complex (see Section 3.1 for more details). The only simplices in K which might intersect any potential second coface f_2 are those in the neighborhood of the dangling edge e. As a result, we filter both the set of samples $\{1, \ldots, n\}$ and the simplicial complex K, to get a subset of samples of Γ and a subcomplex L which are in the neighborhood of e. We cannot, in any straightforward manner, use a distance criterion to detect

```
RECONSTRUCT({x_i}_{i=1}^n)
  (e, f_1, t_1, K) = INITIALIZE({x_i}_{i=1}^n)
  do
    b = barycentre(e)
    (Γ, L) = FILTER({x_i}_{i=1}^n, t_1, b, e, K)
    {u_i}_{i∈Γ} = PROJECT({x_i}_{i=1}^n, e, t_1, b, Γ)
    d_min = ∞
    for all t_2 ∈ Γ
      f_2^pot = e ∪ {t_2}
      if ‖x_{t_2} − b‖ < d_min
        if INTERSECT(f_2^pot, L, {u_j}_{j∈Γ}) = False
          d_min = ‖x_{t_2} − b‖
          f_2 = f_2^pot
    K = K ∪ {σ: σ ⊂ f_2}
    draw (e, f_1, t_1) from K with #cofaces(e) = 1
  until there is no e
  return K
```

Fig. 1. Incremental reconstruction.

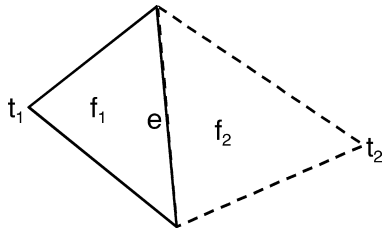

Fig. 2. Adding a simplex. f_1 is the existing coface of the dangling edge e; we wish to add a second coface f_2. The notation in this figure is used throughout the paper.

the neighborhood of e, as the points may be quite non-uniformly sampled. Instead, we use a tangent-space based method.

- *Projection*: Once we are given the subcomplex which is in the neighborhood of the dangling edge e, we can project this subcomplex into the plane. The reason we can do so is that the subcomplex is, topologically, a set of 2-balls, 1-balls, and 0-balls (some of these may be irrelevant—see the discussion in Section 3.3). Furthermore, based on the way we perform the filtering to get the subcomplex L, the projection is close to an isometry on the filtered points. As a result, the problem of finding a second coface f_2 reduces to the problem of adding a triangle to a partially completed triangulation of the plane.
- *Triangulation*: The goal is to find a vertex t_2 in the projected plane, which when combined with the dangling edge e, produces an appropriate second coface f_2 for e (see Fig. 2). This vertex t_2 (which belongs to the filtered subset of samples Γ) may either be an isolated vertex, or may belong to an edge or a triangle. In any case, the key is to for the new triangle f_2 not to intersect any other simplex in the subcomplex L, except in the natural ways allowed for simplicial complexes. An algorithm is given for determining whether or not simplices intersect. Of all of the potential second cofaces f_2 for the dangling edge e, we choose the "best" one: the one which is geometrically closest to the barycentre of e. While this criterion is not the only possible criterion, it is natural and works well in practice.

Each of these steps will be now be fleshed out in greater detail. But first, we briefly digress to introduce some notation with respect to simplicial complexes.

3.1. Simplicial complex notation

Some notation from the theory of simplicial complexes will be used. In general, the simplicial complexes that are referred to in the pseudocode (K and L) are *abstract* simplicial complexes, and the simplices (0-simplices t_1 and t_2, 1-simplex e, and 2-simplices f_1 and f_2) are *abstract* simplices. An abstract simplicial complex is a set V, together with a collection K of subsets of V, called abstract simplices, such that:

1. For all $v \in V$, $\{v\} \in K$. The sets $\{v\}$ are referred to as the vertices of K.
2. If $\sigma \in K$ and $\tau \subset \sigma$, then $\tau \in K$.

Generally, we will refer to the set K by itself as the simplicial complex.

We may now turn to *geometric* simplicial complexes. A d-simplex is given by $\sigma = conv(p_0, \ldots, p_d)$, where the set of points $\{p_i\}_{i=0}^{d}$ are affinely independent, and *conv* indicates the convex hull. A face of σ is given by $\tau = conv(q_0, \ldots, q_c)$ where $\{q_j\}_{j=0}^{c} \subset \{p_i\}_{i=0}^{d}$; similarly, σ is a coface of τ. We will sometimes write $\tau \leqslant \sigma$. A geometric simplicial complex Σ is a finite set of simplices such that

1. $\sigma \in \Sigma$ and $\tau \leqslant \sigma \Rightarrow \tau \in \Sigma$ and
2. $\sigma, \sigma' \in \Sigma \Rightarrow \sigma \cap \sigma' \leqslant \sigma, \sigma'$ or $\sigma \cap \sigma' = \emptyset$.

In other words, all of the simplices must intersect in "natural" ways, or not at all.

The algorithm will return an abstract simplicial complex K; with the natural isomorphism which maps integers i to the their corresponding samples x_i, we hope to also have a geometric simplicial complex. We qualify the previous sentence with the word "hope", as we might get non-simplicial behavior, analogous to an immersion, rather than an embedding of a manifold. Experimentally, it seems that we get properly embedded simplicial complexes in the end.

Note finally that we will sometimes use notation interchangeably, for example referring to an abstract 2-simplex as a triangle. Also, we will use superscripts to denote elements of a particular simplex: for example, e^1 and e^2 denote the first and second vertices of the edge e.

3.2. Initialization

In order for an incremental approach to work, there must be a way of choosing an initial simplex; any face of that simplex may then be taken as the initial dangling edge e. Our method of initialization is shown in pseudocode in Fig. 3.

INITIALIZE($\{x_i\}_{i=1}^n$)
  $q = 1$
  $r = \text{argmin}_{j \neq q} \|x_j - x_q\|$
  $J = \{j \neq q, r\text{: angle}(x_j - x_q, x_r - x_q) \geqslant \pi/4\}$
  $s = \text{argmin}_{j \in J} \|x_j - x_q\|$
  $f = \{q, r, s\}$
  $e = \{r, s\}$
  $t_1 = s$
  $K = K \cup \{\sigma : \sigma \subset f_1\}$
  return (e, f_1, t_1, K)

Fig. 3. Initialization: finding the first face and "dangling edge".

FILTER($\{x_i\}_{i=1}^n, t_1, b, e, K$)
  $\rho = \frac{x_{e1} - b}{\|x_{e1} - b\|}$
  $\xi_1 = (x_{t_1} - b) - \langle x_{t_1} - b, \rho \rangle \rho$
  $\Gamma = \emptyset$
  for $i = 1$ to n
    $\xi_2 = (x_i - b) - \langle x_i - b, \rho \rangle \rho$
    if angle$(\xi_1, \xi_2) \geqslant \pi - \varepsilon$
      $\Gamma = \Gamma \cup \{i\}$
  $L = \{\sigma \in K\text{: vert}\{\sigma\} \subset \Gamma\}$
  return (Γ, L)

Fig. 4. Filtering of the simplicial complex K.

A simple method for choosing the first simplex is to take an arbitrary vertex (in our case, vertex number 1) as the first vertex of the simplex, and take the two vertices which are closest to the first vertex as the second and third vertices of the simplex. The problem with this simple algorithm is that the plane running through the simplex so chosen may be very different from the tangent spaces of any of the three points. For example, suppose that a surface embedded in $\mathbb{R}^3$ is given locally by the paraboloid $z = x^2 + y^2$. Suppose further that vertex number 1 is $q = (0, 0, 0)$, while the two closest vertices to this vertex are $r = (0, 0.1, 0.01)$ and $s = (0, -0.1, 0.01)$. The plane running through a simplex composed of these three vertices is parallel to the y–z plane; however, the tangent spaces of each of the three vertices are nearly (if not exactly) parallel to the x–y plane.

The problem in the example above is the vectors $x_r - x_q$ and $x_s - x_q$ are nearly linearly dependent. In order to remedy this situation, the following simple modification is made. The first vertex q is still chosen arbitrarily, and the second vertex r is still taken to be the closest vertex to q. However, the third vertex s is the closest vertex to q such that the vectors $x_r - x_q$ and $x_s - x_q$ form an angle of at least $\pi/4$. This forces the vectors to be *numerically* linearly independent; as a result, the plane running through the simplex should be a good approximation to the tangent spaces at any of the three vertices, assuming dense enough sampling.

A somewhat related co-cone based algorithm is used in [16]; however, this algorithm requires the calculation of the Voronoi Diagram of the samples, which is computationally very costly when the dimension of the embedding space is high.

3.3. Filtering

Before the second coface f_2 is added, we have a simplicial complex K. We would like to ensure that after adding f_2, we still maintain a simplicial complex; that is f_2 can only intersect the other simplices of K is the "natural" ways allowed for a simplicial complex (see Section 3.1 for more details). The only simplices in K which might intersect any potential second coface f_2 are those in the neighborhood of the dangling edge e. As a result, we filter both the set of samples $\{1, \ldots, n\}$ and the simplicial complex K, to get a subset of samples of Γ and a subcomplex L which are in the neighborhood of e.

We cannot, in any straightforward manner, use a distance criterion to detect the neighborhood of e, as the points may be quite non-uniformly sampled. Instead, we use a tangent space based method. The idea is to approximate the tangent space of points near the dangling edge e by the plane P_1 running through e's first coface f_1. Now, consider the vertex i; we may combine this vertex with the dangling edge e, to form a second plane P_2. The idea is that we consider i to lie in the neighborhood of the dangling edge if the two planes P_1 and P_2 are almost coincident, that is, there is an angle between them of nearly π; see Fig. 5. The intuition behind this filter is straightforward: if the original surface M was smooth and well-sampled, there should be several points i around any dangling edge e such that the two planes P_1 and P_2 form an angle of greater than $\pi - \varepsilon$. The size of ε reflects the fact that we are not infinitely well-sampled; we have found $\varepsilon = \pi/4$ to be an excellent choice in practice.

A question remains: how do we actually find the angle between the planes P_1 and P_2? This angle is computed as the angle between two vectors ξ_1 and ξ_2; ξ_i lies in plane P_i, and is perpendicular to the edge e. See Fig. 4 for precise details.

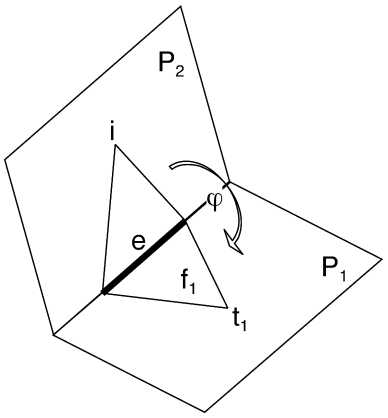

Fig. 5. The point i is retained if the angle ϕ is close enough to π.

PROJECT($\{x_i\}_{i=1}^n, e, t_1, b, \Gamma$)
$T =$ orthonormal basis for $\{x_{e2} - x_{t_1}, x_{e1} - x_{t_1}\}$
for $i \in \Gamma$
$u_i = \frac{\|x_i - b\|}{\|T^T(x_i - b)\|} T^T(x_i - b)$
return $\{u_i\}_{i \in \Gamma}$

Fig. 6. Projection onto the plane.

Once the points have been filtered, it is a simple matter to filter the simplices. If K is the original simplicial complex, and Γ is the filtered set of vertices, then $L = \{\sigma \in K\colon \text{vert}\{\sigma\} \subset \Gamma\}$ is the filtered subcomplex. In other words, we retain exactly those simplices in which all of the vertices have passed the angle test. Now, we may ask the question: are there any simplices which are not in the neighborhood of the dangling edge which are retained? The answer is yes. Such a simplex comes about if it lies, approximately, in the tangent space of the first coface f_1 of the dangling edge e. Note, however, that regardless of the location of such a simplex, it is distorted very little in the process of projection, to be discussed in the next section. As a result, there are two possibilities: (a) the simplex is in the neighborhood of the dangling edge, in which case it is crucial to the incremental triangulation procedure; (b) the simplex is not in the neighborhood of the dangling edge, and is not distorted, in which case it does not affect the incremental triangulation procedure.

3.4. Projection

The ultimate goal is to choose a second coface f_2 for the dangling edge e. In order to do so, we would like to be able to perform "incremental triangulation" in the plane, which will be discussed in the next section. However, incremental triangulation is only possible if we situate the neighborhood of e, as embodied in the filtered subcomplex L, in the plane. Thus, we project all points in Γ and simplices in L, retaining the same abstract simplicial (sub)complex, but obtaining a new geometric one.

Let T be an orthonormal basis for the plane running through the first coface f_1 of the dangling edge; T is a $D \times 2$ matrix. In this, case a straightforward orthogonal projection of a point x onto this plane is specified by

$$\tilde{u} = T^T(x - b),$$

where b is the barycentre of the dangling edge. This particular choice makes the barycentre b the origin of the plane; this is an arbitrary, but sensible, choice given the crucial role of the dangling edge.

Because of the filtering, all points lie near the plane onto which we are projecting; thus, the projection is close to an isometry. However, whatever local distortion is introduced may be lessened if we modify the projection slightly. In

INTERSECT(f_2^{pot}, L, $\{u_j\}_{j\in\Gamma}$)
  test = False
  for all $\sigma \in L$ and while test = False
    solve the following linear program:

    $\max_{\alpha,\beta} \theta = \sum_{i\in f_2^{pot}-\sigma} \alpha_i$

    subject to: $\sum_{i\in f_2^{pot}} \alpha_i u_i = \sum_{j\in\sigma} \beta_j u_j$
    $\sum_{i\in f_2^{pot}} \alpha_i = 1, \sum_{j\in\sigma} \beta_j = 1$
    $\alpha_i \geqslant 0, \beta_j \geqslant 0$

    if there is a feasible solution with $\theta^* > 0$
      test = True
  return test

Fig. 7. Determining whether or not the face f_2^{pot} intersects the subcomplex L.

particular, we ensure that the projection of a given sample is the same distance away from the origin of the projected space (which corresponds to the projection of b) as the sample is from b in the embedding space. Thus, we have that

$$u = \frac{\|x-b\|}{\|T^T(x-b)\|} T^T(x-b).$$

(Note that we need not worry about the case $T^T(x-b) = 0$. The reason is that we have filtered all points for which such a relation is possible.)

3.5. Incremental triangulation

We may now turn the task of determining the second coface f_2 of the dangling edge e. In particular, we must find a vertex t_2 in the filtered set of vertices Γ; f_2 is then the 2-simplex whose vertices are the vertices of e combined with t_2. Now, we may form the abstract collection of subsets $\hat{L} = L \cup \{\sigma\colon\ \sigma \subset f_2\}$; furthermore, we may form the collection of subsets in the plane $\hat{G}$ by mapping each vertex i in $\hat{L}$ to its corresponding projected point u_i. The key property for any f_2 to satisfy is that $\hat{G}$ must be a geometric simplicial complex. In other words, f_2 must only intersect the existing simplices in the neighborhood of the dangling edge e in the natural ways which leave the entire ensemble a simplicial complex.

Our first task must be to determine whether such a coface f_2 always exists. Fortunately, this is the case, as can easily be demonstrated; see [25] for a proof. However, note that establishing that the coface f_2 exists does *not* imply that the algorithm constructs a surface. As we have already noted, we do not attempt to prove this (and it is quite possible that it may not be provable).

Now, there may be several triangles f_2 which are valid; not all, of them, however, are "geometrically nice". For example, there could be a very long, skinny triangle which is almost parallel to the dangling edge. Thus, we find all possible triangles which are valid, and choose the one which is best. There are a variety of criteria we could use to designate the best triangle, but we choose a natural one: the proximity of third vertex t_2 to the barycentre b of the dangling edge.

The problem therefore reduces to finding an algorithm for determining whether two simplices, σ (any simplex in L) and f_2^{pot} (a potential second coface) intersect in a "problematic" way. This algorithm can be designed as a linear program:

$$\max_{\alpha,\beta} \theta = \sum_{i\in f_2^{pot}-\sigma} \alpha_i$$

subject to

$$\sum_{i\in f_2^{pot}} \alpha_i u_i = \sum_{j\in\sigma} \beta_j u_j,$$

$$\sum_{i \in f_2^{pot}} \alpha_i = 1, \quad \sum_{j \in \sigma} \beta_j = 1,$$
$$\alpha_i \geqslant 0, \quad \beta_j \geqslant 0.$$

Here, the variables are $\alpha = \{\alpha_i\}_{i \in f_2^{pot}}$ and $\beta = \{\beta_j\}_{j \in \sigma}$; these are the barycentric coordinates of f_2^{pot} and σ, respectively. The three constraints ensure that the two simplices intersect. The objective function will be positive if the two simplices intersect, but do not intersect in a common face; if they intersect in a common face it will be 0. As a result, we simply test if the linear program has a feasible solution with a positive value for the objective function ($\theta^* > 0$). If this is the case, then the simplices intersect in a problematic way, and f_2^{pot} is not a valid second coface.

3.6. Complexity

The complexity of the algorithm can be analyzed directly from Fig. 1. If F is the number of faces of the surface, E is the number of edges, and n is the number of sample points, then:

- INITIALIZE is $O(n)$.
- The main loop runs F times.
- The main loop of FILTER is $O(n)$, and the computation of L is $O(n + E + F)$.
- PROJECT is $O(|\Gamma|)$.
- The scheme to determine simplex–simplex intersection is $O(1)$, as the dimension of the linear program is a constant, unrelated to n, E, or F. As a result, INTERSECT is $O(|L|)$, and the inner loop ("for all $t_2 \in \Gamma$") is $O(|\Gamma||L|)$.
- Drawing a new dangling edge from K takes $O(|K|) = O(n + E + F)$.

Thus, the total complexity is $O(F(n + E + F + |\Gamma||L|))$.

We can simplify this expression, based in part on the fact that K is a manifold. This implies that there are exactly two cofaces for each edge; since each triangle has three edges, we must have that the number of edges is $E = 3F/2$. Furthermore, we have that the Euler Characteristic is given by $\chi = n - E + F = n - F/2$; assuming that the genus is a constant independent of the number of samples n, then so is χ (which depends linearly on the genus), and thus $F = O(n)$. This allows us to reduce the complexity to $O(n^2 + n|\Gamma||L|)$. It is possible that $|\Gamma| = |L| = O(n)$; this can happen if there is a very large flat area of the manifold. In this case, the complexity is $O(n^3)$. In more typical cases, however, $|\Gamma| = |L| = O(1)$; this occurs when the neighborhoods are much more localized. In this case, the complexity is $O(n^2)$.

The key is that the exponent of n is independent of D, the dimension of the embedding space. This comes about because we do not use a partition of the entire space (such as a Voronoi Diagram) to reconstruct the manifold; the incremental algorithm works the same way in any space.

4. Results

Some results of the incremental reconstruction algorithm are shown. Fig. 8 shows the reconstruction of a non-uniformly sampled sphere; the high degree of non-uniformity, illustrated in detail in Fig. 9, was generated by choosing the sample points entirely at random. Fig. 10 shows another example of a surface which is a topological sphere. This surface was constructed by deforming a sphere under a particular force field, using a partial differential equation. It is also possesses non-uniformly distributed points; in this case, the source of the non-uniformity is the marching cubes algorithm, which is used to extract the surface.

Next, we turn to some surfaces which are topologically different from a sphere. The simplest example of such a surface is a torus, shown correctly reconstructed in Fig. 11.

A more complex object, with genus 5, is shown in Fig. 12. This surface was constructed by beginning with a smoothed out cube, and drilling holes through all three pairs of opposite faces. As in the example of Fig. 10, the surface is generated from an implicit representation using the marching cubes algorithm, leading to a non-uniform distribution of samples.

Thus far, we have seen surfaces which are "pedestrian" by the standards of surface reconstruction algorithms. Despite the reasonably complex topology of the last example, we would expect any of a variety of the algorithms

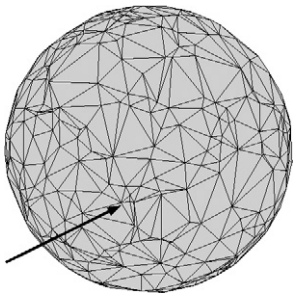

Fig. 8. A sphere.

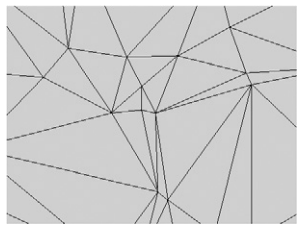

Fig. 9. A detail of the reconstruction from Fig. 8.

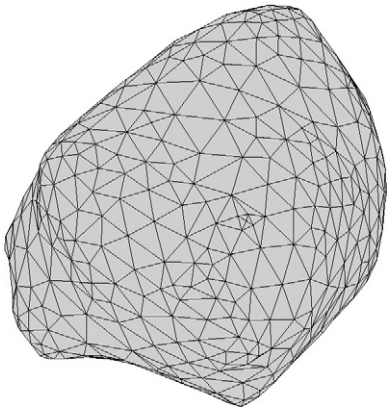

Fig. 10. A topological sphere.

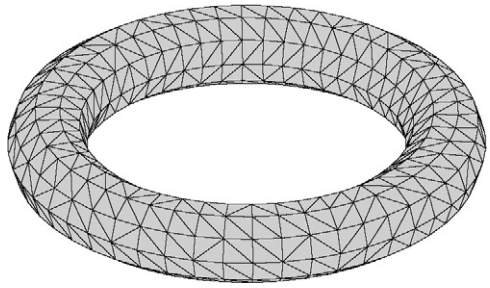

Fig. 11. A torus.

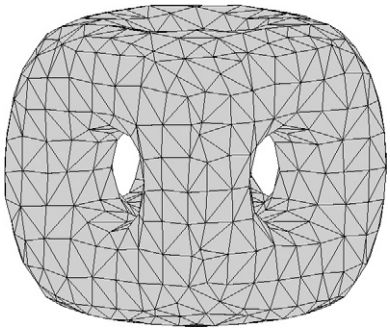

Fig. 12. A surface with genus 5.

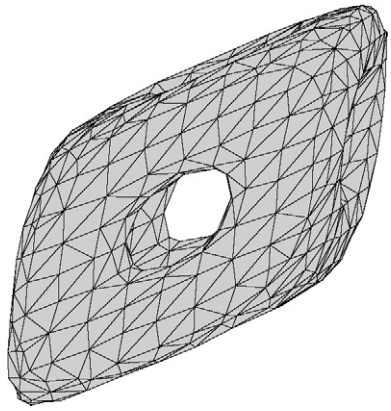

Fig. 13. A three-dimensional slices of a surface embedded in $\mathbb{R}^{40}$.

discussed in Section 2 to be able to handle these surfaces. Let us turn now to surfaces which cannot be handled by these algorithms, namely, surfaces which are embedded in embedding spaces of dimension greater than 3. To begin with, we embedded the "six hole" surface shown in Fig. 12 and through an isometry into $\mathbb{R}^{40}$. In particular, each point was transformed according to $x' = \Theta x$, where $\Theta \in \mathbb{R}^{40\times 3}$ satisfying $\Theta^T\Theta = I$. As expected, the reconstruction was correct, and matched the reconstruction of the original surface exactly. A slice of this surface, along dimensions 8, 13, and 25, is shown in Fig. 13. Note that the surface looks sheared, due to the fact that it has been sliced along three dimensions; projection along these dimensions does not preserve angles, despite the fact that the overall transformation is an isometry.

Perhaps more interesting than this artificial example are surfaces which can *only* be embedded in $\mathbb{R}^4$. The most famous of these is the Klein Bottle, which cannot be embedded in $\mathbb{R}^3$ due to its non-orientability. Two different embeddings of the Klein Bottle are given. The first embedding is based on the classic "figure eight" immersion in $\mathbb{R}^3$, which is given by

$$
\begin{aligned}
x_1(u,v) &= \big(a + \cos(u/2)\sin(v) - \sin(u/2)\sin(2v)\big)\cos(u),\\
x_2(u,v) &= \big(a + \cos(u/2)\sin(v) - \sin(u/2)\sin(2v)\big)\sin(u),\\
x_3(u,v) &= \sin(u/2)\sin(v) + \cos(u/2)\sin(2v),
\end{aligned}
$$

where $u \in [0, 2\pi)$, $v \in [0, 2\pi)$. It is possible to turn this immersion into an embedding in $\mathbb{R}^4$; however, we choose instead to use a very simple embedding into $\mathbb{R}^5$, taking $x_4(u,v) = \sin(u)$ and $x_5(u,v) = \cos(v)$. With a small amount of inspection, it is easy to see that the self-intersections in $\mathbb{R}^3$ are removed with the addition of these two extra coordinates. The correct reconstruction of this surface is shown from two points of view in Figs.14 and 15; in both cases, the three-dimensional slice is along the first three dimensions. The self-intersections are clearly visible.

A second, more unusual embedding of the Klein Bottle into $\mathbb{R}^5$ is suggested by Barrett O'Neill [26]. The embedding

$$
\begin{aligned}
x_1(u,v) &= \cos(u)\cos(v),\\
x_2(u,v) &= \sin(u)\cos(v),\\
x_3(u,v) &= 2\cos(u/2)\sin(v),\\
x_4(u,v) &= 2\sin(u/2)\sin(v),\\
x_5(u,v) &= \cos(v),
\end{aligned}
$$

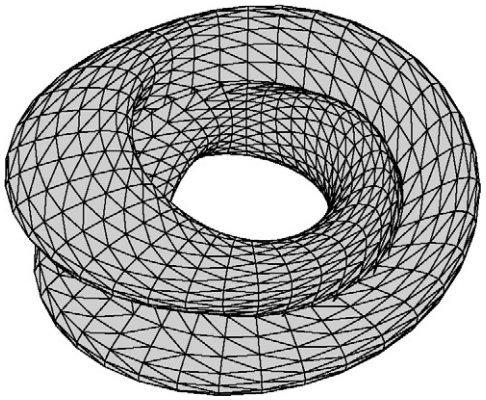

Fig. 14. The "figure 8" Klein Bottle.

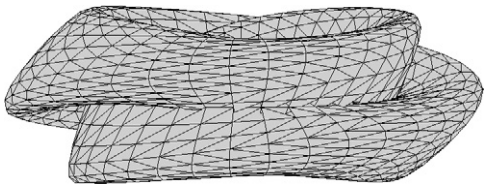

Fig. 15. Another view of the "figure 8" Klein Bottle.

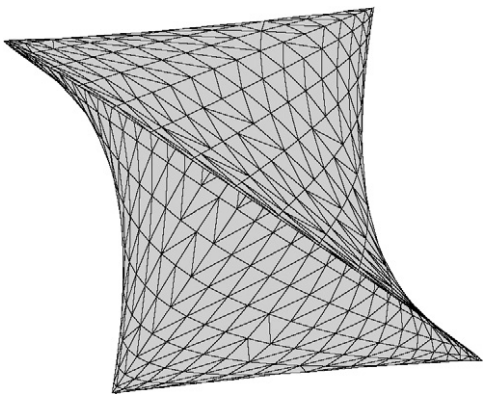

Fig. 16. O'Neill's flat Klein Bottle: dimensions 1, 4, 5.

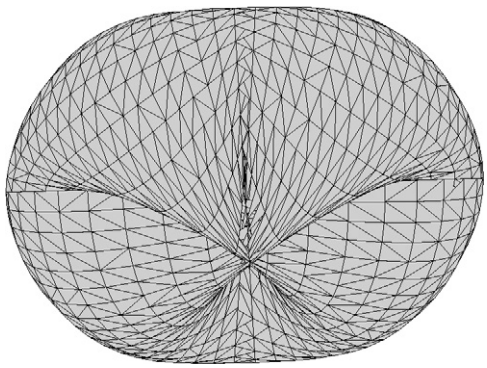

Fig. 17. O'Neill's flat Klein Bottle: dimensions 2, 3, 4.

where $u \in [0, 2\pi)$, $v \in [0, 2\pi)$, represents a *flat* Klein Bottle, in that it has zero Gaussian curvature everywhere. Once again, the correct reconstruction is attained, and slices along several different combinations of three dimensions are shown in Figs. 16, and 17. Again, the self-intersections are clearly visible.

Acknowledgements

This work was supported in part by the US National Science Foundation, under Award IIS-0133144.

References

[1] D. Freedman, Efficient simplicial reconstructions of manifolds from their samples, IEEE Trans. Pattern Analysis and Machine Intelligence 24 (10) (2002) 1349–1357.
[2] J.D. Boissonnat, Geometric structures for three-dimensional shape reconstruction, ACM Transactions on Graphics 3 (1984) 266–286.
[3] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, Surface reconstruction from unorganized points, in: Proc. SIGGRAPH'92, 1992, pp. 71–78.
[4] H. Hoppe, T. DeRose, T. Duchamp, H. Jin, J. McDonald, W. Stuetzle, Piecewise smooth surface reconstruction, in: Proc. SIGGRAPH'94, 1994, pp. 19–26.
[5] B. Curless, M. Levoy, A volumetric method for building complex models from range images, in: Proc. SIGGRAPH'96, 1996, pp. 303–312.
[6] F. Bernardini, C.L. Bajaj, Sampling and reconstructing manifolds using α-shapes, in: Proc. 9th Canadian Conf. on Comput. Geom., 1997, pp. 193–198.
[7] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, G. Taubin, The ball-pivoting algorithm for surface reconstruction, IEEE Trans. Vis. 5 (4) (1999) 349–359.
[8] N. Amenta, M. Bern, Surface reconstruction by Voronoi filtering, Discrete and Computational Geometry 22 (1999) 481–504.
[9] N. Amenta, S. Choi, T.K. Dey, N. Leekha, A simple algorithm for homeomorphic surface reconstruction, in: Proc. 16th ACM Sympos. Comput. Geom., 2000, pp. 213–222.
[10] N. Amenta, S. Choi, R.K. Kolluri, The power crust, union of balls, and the medial axis transform, Computational Geometry: Theory and Applications 19 (2001) 127–153.
[11] T.K. Dey, S. Funke, E. Ramos, Surface reconstruction in almost linear time under locally uniform sampling, in: Proc. 17th European Workshop on Comput. Geom., 2001.
[12] S. Funke, E. Ramos, Smooth-surface reconstruction in near-linear time, in: Proc. 13th ACM–SIAM Sympos. Discrete Algorithms, 2002, pp. 781–790.
[13] T.K. Dey, J. Giesen, J. Hudson, Delaunay based shape reconstruction from large data, in: Proc. IEEE Symposium in Parallel and Large Data Visualization and Graphics, 2001, pp. 19–27.
[14] T.K. Dey, J. Giesen, Detecting undersampling in surface reconstruction, in: Proc. 17th ACM Sympos. on Comput. Geom., 2001, pp. 257–263.
[15] T.K. Dey, J. Giesen, N. Leekha, R. Wenger, Detecting boundaries for surface reconstruction using co-cones, Internat. J. Computer Graphics and CAD/CAM 16 (2001) 141–159.
[16] T.K. Dey, J. Giesen, S. Goswami, W. Zhao, Shape dimension and approximation from samples, in: Proc. 13th ACM–SIAM Sympos. Discrete Algorithms, 2002, pp. 772–780.
[17] M. Gopi, S. Krishnan, C.T. Silva, Surface reconstruction based on lower dimensional localized Delaunay triangulation, Computer Graphics Forum (EUROGRAPHICS 2000) 19 (3) (2000) C467–C478.
[18] M. Gopi, S. Krishnan, A fast and efficient projection-based approach for surface reconstruction, in: Proc. 15th Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'02), 2002.
[19] D. Cohen-Steiner, F. Da, Greedy Delaunay based surface reconstruction algorithm, Rapport de recherche RR-4564, INRIA–Sophia Antipolis, 2002.
[20] P. Crossno, E. Angel, Isosurface extraction using particle systems, in: Proc. IEEE Visualization '97, 1997, pp. 495–498.
[21] R. Mencl, H. Muller, Interpolation and approximation of surfaces from three dimensional scattered data points, in: State of the Art Reports, Eurographics, 1998, pp. 51–67.
[22] U. Adamy, J. Giesen, M. John, New techniques for topologically correct surface reconstruction, in: Proc. of the 11th Annual IEEE Visualization Conference, 2000, pp. 373–380.
[23] J.D. Boissonnat, F. Cazals, Smooth surface reconstruction via natural neighbour interpolation of distance functions, Computational Geometry: Theory and Applications 22 (1) (2002) 185–203.
[24] H. Edelsbrunner, Surface reconstruction by wrapping finite point sets in space, in: B. Aronov, S. Basu, J. Pach, M. Sharir (Eds.), Ricky Pollack and Eli Goodman Festschrift, Springer-Verlag, 2002.
[25] D. Freedman, Surface reconstruction, one triangle at a time, in: Proc. 16th Canadian Conf. on Comput. Geom. (CCCG), 2004, pp. 15–19.
[26] B. O'Neill, http://www.math.ucla.edu/~bon/tompkins.html.