Efficient Simplicial Reconstructions of Manifolds from Their Samples

Daniel Freedman, *Member*, *IEEE*

Abstract—A new algorithm for manifold learning is presented. Given only samples of a finite-dimensional differentiable manifold and no a priori knowledge of the manifold's geometry or topology except for its dimension, the goal is to find a description of the manifold. The learned manifold must approximate the true manifold well, both geometrically and topologically, when the sampling density is sufficiently high. The proposed algorithm constructs a simplicial complex based on approximations to the tangent bundle of the manifold. An important property of the algorithm is that its complexity depends on the dimension of the manifold, rather than that of the embedding space. Successful examples are presented in the cases of learning curves in the plane, curves in space, and surfaces in space; in addition, a case when the algorithm fails is analyzed.

Index Terms—Machine learning, differentiable manifold, simplicial complex.

1 Introduction

This paper is concerned with the problem of *manifold learning*. Given a sampling of points drawn from a compact differentiable manifold, the goal is to find a description of the manifold. More formally, the problem may be posed as follows:

Manifold Learning Problem. Given a dense sampling X of some K-dimensional compact C^∞ manifold F which is embedded in a Hilbert space Z, construct a K-dimensional compact manifold R which approximates F well.

Such a formulation is not complete in the sense that many concepts, such as "good approximation" and "dense sampling," remain vague; these issues will be cleared up in Section 1.2. For the moment, let us focus on practical applications of manifold learning in order to illustrate the type of problem we wish to treat.

1.1 Applications

One class of algorithms, drawn from the computer vision literature, for which manifold learning is useful is the class of contour trackers. A contour tracker seeks to follow an object as it moves through a video stream by following its contour, i.e., the curve which represents its silhouette; examples include [1], [2], [3], [4]. It is often useful, and sometimes necessary, for such trackers to have an a priori model of the dynamics of the object to be tracked. Such a model may occasionally be known or derived from first principles, but, in the general case, it is learned from training data. The model can be specified as a general stochastic dynamical system in the space of curves; however, in many cases, it is of great benefit to first restrict the possible geometries of an object's contour from the entirety of curve space to a particular subset, often labeled the "shape space." For instance, if we wish to track the movement of a particular human organ for a medical application, we will find that our algorithm will profit tremendously from the knowledge that such an organ has a fixed range of possible appearances and never looks like, say, the large intestine. One method for acquiring this information, i.e., for learning the shape space, is that of manifold learning; such a scheme will succeed when the training set is a dense enough sampling of the underlying manifold. Finally, given the shape space, the dynamical model may be learned.

Manifold learning might also be employed in recognition schemes. In order to properly identify an object within an image, one often needs a good description of the class from which the object was drawn. Such a class may be well characterized by a manifold within the "space of objects," where the precise mathematical meaning of object space may vary between recognition applications. Given a proper training set, manifold learning presents a way of constructing a characterization of the object class.

Additionally, there are a variety of applications for manifold learning which are not ordinarily construed as learning problems. One major example is that of surface reconstruction, which is useful in CAD and computer graphics; in such a problem, points in $\mathbb{R}^3$ are given and the desired output is a surface. One use for such an algorithm is "reverse engineering," where a machine part is scanned with a laser range finder and a model of the object's geometry is automatically generated. The latter scheme is also pertinent in the field of graphics and allows for graphical models to be constructed directly from real objects without modeling.

1.2 Basic Aspects of the Problem

Having posed the problem, some of its challenges may now be discussed. The key problem in designing a manifold learning algorithm is that the sample points X are *unorganized*. That is, no relationship among the points is

• *The author is with the Department of Computer Science, Rensselaer Polytechnic Institute, Amos Eaton 205, Troy, NY 12180-3590. E-mail: freedman@cs.rpi.edu.*

Manuscript received 27 June 2001; revised 28 Jan. 2002; accepted 27 Feb. 2002.
Recommended for acceptance by D. Jacobs.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 114444.

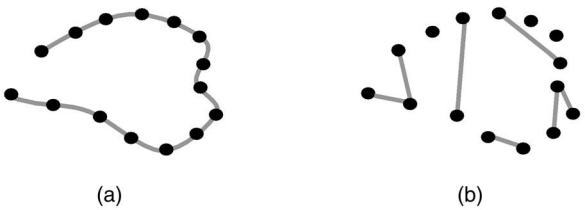

Fig. 1. Reconstruction from unorganized points. (a) The original curve with some sample points. Given the original, the reconstruction looks "obvious." (b) Since the points are unorganized, it isn't clear which should be joined together. Some possibilities are shown which are clearly incorrect given knowledge of the original.

known beforehand. This difficulty is seen most easily in the context of what might seem, at first blush, a simple problem: trying to "learn a curve" embedded in $\mathbb{R}^2$ from its samples. If the points are organized, i.e., the adjacency relationship between samples is known, then reconstruction is completely straightforward. This situation isn't particularly interesting because it is quite obvious that the more finely sampled the true manifold is, the closer the reconstruction will be to the original; this is just a matter of calculus. However, suppose that the adjacency relationship among the points is *not* known, as is illustrated in Fig. 1. In this case, the issue of how to learn the curve is not at all clear. Should all samples be attached to their nearest neighbors? This would leave holes, as in Fig. 2. Instead, one might begin by connecting a single point to its nearest neighbor and then connecting that point to is nearest neighbor except for the one already attached, etc.; in this case, one runs into the problem shown in Fig. 2. It is fairly easy to think of counterexamples to such simple greedy schemes. Furthermore, we have been looking at the "relatively straightforward" case provided by curves embedded in two dimensions. What of more complex cases, such as 7-manifolds embedded in $\mathbb{R}^{82}$ or K-manifolds embedded in the space of curves? In this case, it is not even particularly easy to think of heuristic algorithms like the one mentioned above. As a result, sophisticated algorithms which possess mathematical properties which are amenable to analysis must be elaborated if there is to be a chance of attacking the manifold learning problem.

In this context, we may return to the problem statement given in Section 1 and clarify some relevant issues. First, the manifold R is said to approximate the manifold F well if

- R is homeomorphic to F and
- $d(R, F)$ is sufficiently small, where d is an appropriate metric, such as the symmetric Hausdorff distance:

$$d(R,F) = \max_{f\in F}\left[\min_{r\in R}\|f-r\|\right] + \max_{r\in R}\left[\min_{f\in F}\|f-r\|\right].$$

Note that diffeomorphism between R and F is not required; in fact, the proposed algorithm generates a C^0 manifold, in the form of a simplicial complex. Requiring diffeomorphism makes the problem more difficult.

The second issue that arises in the problem statement, and throughout the paper, concerns the idea of sampling density. There are various possible formalizations of this

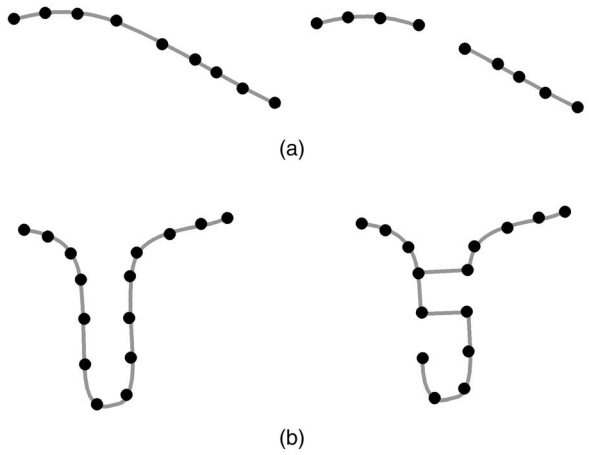

Fig. 2. Bad manifold reconstruction schemes. On the left is the original curve, on the right the reconstruction. (a) Attaching all samples to their nearest neighbors leaves holes. (b) Greedy nearest neighbors (see description in the text) leads to a different kind of problem.

concept; perhaps the most useful definition of sampling density is given in [5] and is based on the notion of local feature size. The local feature size at a given point on the manifold is defined to be the distance from that point to the medial axis of the manifold. This definition captures how close a given point is to other parts of the manifold, where the "other part" may be local (arising from high curvature) or global (for example, arising from a separate connected component). Given this concept of feature size, sampling density can then be defined as follows: A manifold is said to be ρ-sampled, $0 < \rho \leq 1$, if every point on the manifold is at most ρ times the local feature size away from a sample point. As ρ decreases, the manifold becomes sampled increasingly finely. Interested readers are pointed to [5].

This notion of sampling is quite sensible: The idea is to require a higher sampling density in regions where there is a lot of detail and a smaller density in regions where there is not much detail. A manifold learning algorithm should possess the property that, if we are given a ρ-sampling of the true manifold and if $\rho < \rho_0$ for some number $\rho_0 < 1$, then the learned manifold should be a good approximation to the true manifold. Although no proofs will be presented in this paper, this type of criterion may be useful for the reader to bear in mind.

Finally, it is important to address two separate problems which plague any reconstruction algorithm: noise and undersampling. Almost any real data set will be somewhat noisy: The samples will not lie exactly on a manifold, but rather will be close to it. Furthermore, in practice, it is often difficult to guarantee that the manifold will be sampled sufficiently densely everywhere. It is clear that both of these problems are the norm, rather than the exception, when dealing with real data sets. It is therefore important to state, at the outset, that the algorithm presented in this paper *will not attempt to deal with either problem*. The reason for this is straightforward: The problem of manifold reconstruction, given arbitrary (and unknown) geometry and topology, is sufficiently complicated on its own. The design of an algorithm to reconstruct manifolds from samples which are both sufficiently dense and free of noise is a very difficult

challenge. Clearly, the more interesting case (from a practical point of view) occurs when we consider the presence of both noise and undersampling. The algorithm which is described in this paper may be thought of as a first step toward such an algorithm. The ultimate goal would be to extend the results presented here to the more realistic, and yet more challenging, case which involves and undersampling: Such an algorithm would allow for the reconstruction of real data sets. Nonetheless, the algorithm which is described in this paper may be seen as interesting in its own right.

1.3 Outline of the Paper

Section 2 discusses the manifold learning literature and focuses on relevant work from the field of computational geometry. In particular, the methods presented in [6], [5] will be discussed in detail as they represent the state of the art in manifold learning algorithms. They represent a good model for the type of algorithm we seek in that they provide the possibility of capturing the manifold's topological and geometrical details with minimal prior knowledge about the manifold. Algorithmic drawbacks to these methods are then enumerated. Section 3 outlines the new approach. The major novelty of this approach lies in its reliance on tangent space information. The algorithm is fairly complex and, thus, its description is divided into three distinct parts. Section 4 explains the mathematical motivation behind the algorithm and details its complexity as a function of the number of sample points. Finally, Section 5 illustrates several examples of the algorithm in action, discusses its failings, and points to some directions for future research.

2 A Review of Existing Methods

There are a number of techniques for attacking the problem of manifold learning. One class goes under the heading of dimension reduction techniques; a common example is the Karhunen Loeve Transform [7]. The KLT allows for a mapping of example points embedded in a space to a linear manifold (or affine flat) of low dimension. However, the problem is that, unless the original manifold F is linear, the reconstruction R will not satisfy the topological and geometrical properties specified in the problem statement. An example of this phenomenon is a circle embedded in $\mathbb{R}^3$. Clearly, the circle is one-dimensional; however, regardless of whether a one-dimensional (line) or two-dimensional (plane) approximation is used, nothing close to the original is recovered. It is apparent that, no matter how finely sampled the circle is, the KLT will always result in very bad approximations. It may be noted that the KLT is the solution to a problem which tries to find the linear manifold, of a particular dimension, which is such that the sum of the squared distances from the samples to the manifold is minimum.

An obvious extension would involve looking for this minimum over a broader class of manifolds, which could possibly be nonlinear. The objections to this procedure are twofold. First, assumptions still have to be made on the class of manifolds to look at and this assumption will generally prove restrictive. Consider, for example, the case of a 1-manifold embedded in $\mathbb{R}^2$. We may consider a rather general class of curves. But, what if the true manifold is doubly connected? In this case, we should have to broaden the class of curves being considered. But, what if we don't know the connectedness? It is clear that all kinds of difficulties may arise from such a paradigm. The second issue is more practical: Actually minimizing such an objective function is probably difficult. That is, what is nice about the KLT is that finding the global minimum is analytically feasible due to the simple linear nature of the manifolds. Global minimization in a more general case could prove very difficult and local minimization is highly unsatisfying.

A second class of algorithms take a rather different tack. The KLT and related methods are *global* in that they assume a form for the *entire* manifold and try to find the appropriate element of this class based on the sample points; *local* methods make no assumptions on the overall shape of the manifold, but, rather, are sensitive to the geometry and topology of each neighborhood. Examples of this type of approach may be seen in the graphics literature [8], [9], [10], which presents several instances of attempts to reconstruct surfaces (2-manifolds) embedded in $\mathbb{R}^3$. These results, while often impressive in practice, are difficult to analyze mathematically and are therefore not extensible to the general case of manifold learning. Bregler and Omohundro [11] attack the general problem in this local manner; however, they do not generate an actual manifold, but, rather, a function which projects points in the embedding space onto their nearest neighbors in the manifold. Furthermore, their method is problematic in that the object it implicitly generates is not truly a manifold as it may have varying dimension in different parts.

More recently, a version of the manifold learning problem has been formally solved in [6], [12], [5], for the special cases of reconstructing curves (1-manifolds) in $\mathbb{R}^2$ and surfaces (2-manifolds) in $\mathbb{R}^3$. The solutions have used machinery from computational geometry, both in terms of construction and proofs. Other papers in the same vein have followed suit [13], [14], [15], [16], [17], [18], [19], [20]. The *crust* is defined to be the set of edges from the Delaunay Triangulation of the set composed of the samples and their Voronoi vertices such that both vertices of the edge belong to the set of samples. Interested readers are referred to [6], in which it is shown that, if the curve is ρ-sampled (in the sense of Section 1.2) for $\rho < 0.252$, then the crust represents a proper reconstruction of the true curve. The algorithm for surface reconstruction [5] is quite similar to the construction of the crust, with one major difference: The set of Voronoi vertices is replaced by the set of *poles*, which are essentially the two farthest Voronoi vertices of a sample's Voronoi cell, one on each side of the surface. The proofs are more complex in this case, but the result is the same: The new construction is shown to be homeomorphic to the original manifold and close to it if the surface is sampled sufficiently finely to begin with.

There are several reasons why these algorithms, as impressive as they are, are not easily extensible to the general case, but one particular reason stands out. This relates to complexity considerations. The Voronoi Diagram of an N point set in $\mathbb{R}^2$ may be calculated in $O(N \log N)$; in $\mathbb{R}^3$, the complexity is $O(N^2)$. However, in the case of $\mathbb{R}^D$, the complexity becomes the critical factor: $O(N^{\lceil D/2 \rceil})$. Since the

embedding spaces we are often interested in may have $D > 100$, this is not a feasible approach. Instead, the complexity should be a function of the dimension of the manifold, K. This is in fact what is achieved in the proposed algorithm.

3 The Algorithm

The method to be considered now will result in a simplicial complex; however, it avoids computing the full Voronoi Diagram/Delaunay Triangulation in order to construct the complex. Instead, it finds approximations to the tangent space at each sample and then reconstructs the manifold based on these tangent spaces. Effectively, the computation is equivalent to calculating something like a partial Delaunay Triangulation, where the triangulation is *restricted to the manifold*. As a result, the complexity of the operation will not depend on the dimension of the embedding space, but rather on the (more geometrically intrinsic) dimension of the manifold.

The following is a brief overview of the algorithm, to be elaborated upon over the next several pages. The algorithm consists of three steps. In the first step, an approximation to the tangent space at each sample is found. Using this information, the second step leads to the calculation of the "local region" at each sample: The local region is essentially the little patch of manifold which surrounds a given sample. Finally, the third step stitches together these local regions into a simplicial complex, which represents the entire manifold. The mathematical motivation for structuring the algorithm in this way is given after the algorithm, in Section 4.1; some readers may find it useful to examine this section concurrently with the algorithm description.

Let us begin by establishing the basic notation. The embedding space is Z, the manifold is F, and the sampling of the manifold is X. $X \subset F \subset Z$, with $|X| < \infty$. Recall that the embedding space is restricted to be a Hilbert space as the use of an inner product is necessary to the algorithm. The goal of the algorithm is to construct a simplicial complex based only on X, denoted $SC(X)$. For each point x, the neighborhood of x, denoted $NBD(x)$, is defined as follows: Out of the entire sampling X, $NBD(x)$ is defined to be the rK closest points to x, where $r \geq 1$ and K is the manifold dimension. r is a parameter which can be set. Given these definitions, the algorithm may now be described. It consists of three steps.

3.1 Step 1: Finding Tangent Spaces

The goal is to find an approximation to the tangent space at each point $x \in X$. Typically, this procedure will only work well for interior points; at points along the boundary, it generally fails. This failure is taken account of explicitly, which will become clearer in the following exposition. In order to find the **approximating tangent space** at x, denoted $ATS(x)$, we must make use of the following definition.

Definition. *A point x is said to be **enclosed** by the $K+1$ points $\{x_i\}_{i=0}^K$ if the projection of x onto the K-dimensional hyperplane running through $\{x_i\}_{i=0}^K$ is contained within the K-dimensional simplex defined by $\{x_i\}_{i=0}^K$.*

It will be useful to have a simple method for testing the enclosure property on a given set of points. Note that the hyperplane running through $\{x_i\}_{i=0}^K$ is given by

$$\mathrm{span}\{x_1 - x_0, \ldots, x_K - x_0\} + x_0.$$

Indeed, any point on the hyperplane may be given by

$$x_0 + \sum_{i=1}^K \lambda_i (x_i - x_0) = x_0 + \sum_{i=1}^K \lambda_i \Omega_i,$$

where $\Omega_i = x_i - x_0$ and λ is a $K \times 1$ column vector which can take on any value in $\mathbb{R}^K$. Now, the projection problem is specified as

$$\min_{\lambda \in \mathbb{R}^K} \left\| x_0 + \sum_{i=1}^K \lambda_i \Omega_i - x \right\|.$$

This can be solved as follows:

$$\begin{aligned} \left\| x_0 + \sum_{i=1}^K \lambda_i \Omega_i - x \right\|^2 &= \sum_{i=1}^K \sum_{k=1}^K \lambda_i \lambda_k \langle \Omega_i, \Omega_k \rangle - \sum_{i=1}^K \lambda_i \langle \Omega_i, \zeta \rangle \\ &\quad - \sum_{i=1}^K \lambda_i \langle \zeta, \Omega_i \rangle + \langle \zeta, \zeta \rangle \\ &= \lambda^T \Psi \lambda - 2\psi^T \lambda + \|\zeta\|^2, \end{aligned}$$

where $\zeta = x - x_0$, Ψ is a Hermitian $K \times K$ matrix given by $\Psi_{ik} = \langle \Omega_i, \Omega_k \rangle$, and ψ is a $K \times 1$ column vector given by $\psi_i = \frac{1}{2}(\langle \Omega_i, \zeta \rangle + \langle \zeta, \Omega_i \rangle)$. This may be now be minimized to yield

$$\lambda^* = \Psi^{-1} \psi.$$

Finally, if we define $\theta_0 = 1 - \sum_{i=1}^K \lambda_i^*$ and $\theta_i = \lambda_i^*$, $i = 1, \ldots, K$, then the condition that x is enclosed by $\{x_i\}_{i=0}^K$ is equivalent to the condition

$$0 \leq \theta_i \leq 1, \quad i = 0, \ldots, K$$

which is quite easy to test.

Given the enclosure property, an algorithm for calculating $ATS(x)$ may now be specified.

Definition. *A set of $K+1$ points $\{x_i\}_{i=0}^K$ which are drawn from the neighborhood of x, $NBD(x)$, is said to be a **tangent basis** for the point $x \in X$ if 1) x is enclosed by $\{x_i\}_{i=0}^K$ and 2) no other point in $NBD(x)$ is enclosed by $\{x_i\}_{i=0}^K$. A shifted version of the hyperplane defined by $\{x_i\}_{i=0}^K$, namely, $\mathrm{span}\{x_1 - x_0, \ldots, x_K - x_0\} + x$, is referred to as x's **approximate tangent space**, or $ATS(x)$. Note that $x \in ATS(x)$.*

It may be the case that a point x has no approximating tangent space; in this case, we declare x to belong to the boundary.

Definition. *Any point x which has no tangent basis is said to belong to the **boundary**, denoted BD.*

The algorithm for finding $ATS(x)$ then follows in a straightforward manner from the definitions. One searches through all $(k+1)$-tuples of points, starting with the nearest first, until one finds a tangent basis. Once the tangent basis is found, $ATS(x)$ is automatically calculated (as in the

definition of the tangent basis). If no tangent basis is found, the point x is declared to belong to the boundary.

Note that this algorithm for calculating the approximate tangent space, as based on the enclosure property, is an intuitive one. The idea is that the ATS is best approximated by a nearby secant, where the correct notion of closeness is given by enclosure; this is essentially inspired by the mean value theorem.

3.2 Step 2: Calculating the Local Region and Edge-Set

The goal is to find the little patch of the manifold which contains x. As stated, this is, of course, ill-defined; however, a particular definition of the local region will be given which is useful in calculating the simplicial complex $SC(X)$.

Definition. *The **perpendicular bisecting halfspace** of the point $x \in X$ with respect to the point $x' \in X$ is denoted by $PBH(x, x')$ and is given by*

$$PBH(x, x') = \{z \in Z : \|z - x\| \leq \|z - x'\|\}.$$

Definition. *The **local region** of $x \in X - BD$, denoted $LR(x)$, is given by*

$$LR(x) = \left[\bigcap_{x' \in X} PBH(x, x')\right] \bigcap ATS(x).$$

Note that the local region is only defined for points that are not in the boundary; that the local region is K-dimensional, as $ATS(x)$ is K-dimensional; and that $x \in LR(x)$ (since $x \in ATS(x), PBH(x, x')$).

Calculating the local region is a matter of deciding which of the constraints provided by intersecting the halfspaces $PBH(x, x')$ are binding. That is, we could write

$$LR(x) = \left[\bigcap_{x' \in ES(x)} PBH(x, x')\right] \cap ATS(x),$$

where $ES(x) \subset X$ and is the smallest such set. $ES(x)$ is referred to as the **edge-set** of x for reasons that will be clear shortly. In order to find the edge-set, a convex hull technique may be used.

In particular, note that

$$ATS(x) = \left\{z \in Z : z = \Omega_x \lambda + x, \lambda \in \mathbb{R}^K\right\},$$

where $\Omega_x \lambda$ is a short form for $\sum_{i=1}^{K} \Omega_{x,i} \lambda_i$ and $\Omega_{x,i}$ are the vectors found in Step 2, and

$$PBH(x, x') = \{z \in Z : \langle \alpha_{x'}, x - \beta_{x'} \rangle \geq 0\},$$

where $\alpha_{x'} = x - x'$ and $\beta_{x'} = \frac{1}{2}(x + x')$. Note that

$$LR(x) = \bigcap_{x' \in X} (PBH(x, x') \cap ATS(x))$$

and

$$\begin{aligned} &PBH(x, x') \cap ATS(x) \\ &= \left\{z \in Z : \langle \alpha_{x'}, \Omega_x \lambda + x - \beta_{x'} \rangle \geq 0, \lambda \in \mathbb{R}^K\right\} \\ &= \left\{z \in Z : \gamma_{x,x'}^T \lambda \geq \delta_{x,x'}, \lambda \in \mathbb{R}^K\right\}, \end{aligned}$$

where $\gamma_{x,x'}$ is a $K \times 1$ column vector whose ith entry is $\langle \Omega_{x,i}, \alpha_{x'} \rangle$, and $\delta_{x,x'} = \langle \alpha_{x'}, \beta_{x'} - x \rangle$ is a scalar. Thus, the problem is reduced to looking for the binding set of inequalities out of the entire set

$$\gamma_{x,x'}^T \lambda \geq \delta_{x,x'} \quad x' \in X - \{x\}.$$

It may be shown that this problem is equivalent to finding the convex hull in $\mathbb{R}^K$ of the points

$$\epsilon_{x,x'} = \frac{\gamma_{x,x'}}{\delta_{x,x'}} \quad x' \in X - \{x\}$$

The constraints which bind are exactly the same as the points which lie on the exterior of the convex hull. In terms of complexity considerations, finding the convex hull N points in $\mathbb{R}^K$ is $O(N^{\lceil K/2 \rceil})$; see [21]. Finding the set of binding constraints gives us $ES(x)$.

Note that finding the local region (and, hence, the edge-set) is equivalent to performing a restricted Delaunay Triangulation calculation. This calculation is made possible through the aforementioned computation of the convex hull.

3.3 Step 3: From the Edge-Set to the Simplicial Complex

The goal is to take the edge-sets $ES(x)\ \forall x \in X - BD$ and to find a simplicial complex. A K-dimensional simplex consists of edges between all pairs of the $K + 1$ points. Thus, an algorithm to convert edge-sets into simplices may proceed as follows.

An undirected graph $G = (V, E)$ is formed as follows: The set of vertices is set to be $V = X$. Given two vertices $x, y \in V$, an edge $e = (x, y)$ is in E iff $x \in ES(y)$ and $y \in ES(x)$. A simplex in the simplicial complex $SC(X)$ if all of the edges of the simplex belong to E. Note that a larger simplicial complex may be formed with a slightly different graph $G' = (V', E')$, where $V' = V$ and $e = (x, y)$ is in E' iff $x \in ES(y)$ *OR* $y \in ES(x)$.

4 Discussion of the Algorithm

The algorithm which was specified in the previous section is fairly involved. We will try to explain aspects of it on two independent fronts: why it works and its complexity.

4.1 Motivation of the Algorithm

The following result is proven in [5]. Let $V(X)$ be the Voronoi Diagram of the set of sample points X of manifold F, where the manifold is of dimension 2 and is embedded in $\mathbb{R}^3$; note that, by $V(X)$ we mean the entire diagram, rather than just the vertices. Let $\Upsilon = V(X) \cap F$; intersecting the Voronoi Diagram, which is a partition of $\mathbb{R}^3$, with the manifold F has the effect of partitioning the manifold. Now, let $\mathrm{Dual}(\Upsilon)$ be the dual of Υ, that is, if two samples x_1 and x_2 are contained in cells of Υ which border each other, let $\mathrm{Dual}(\Upsilon)$ contain an edge between x_1 and x_2. It can be shown that if X is a 0.1-sample of F (in the sense of Section 1.2), $\mathrm{Dual}(\Upsilon)$ will be a simplicial complex which is homeomorphic to F and sufficiently close to it.

Of course, this theorem cannot be used for constructing $SC(X)$ since the construction of $\mathrm{Dual}(\Upsilon)$ relies on knowledge

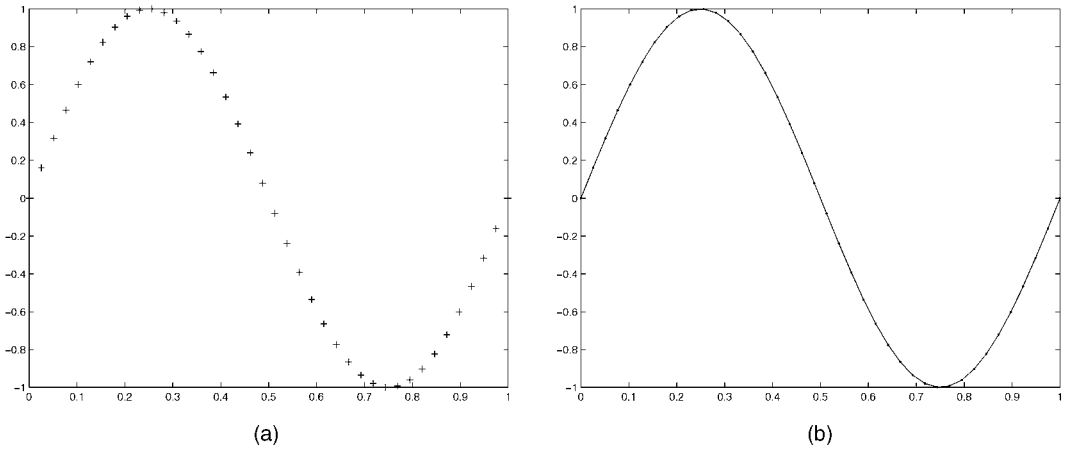

Fig. 3. A simple curve in the plane. (a) The samples. (b) The reconstructed manifold.

of F, which we do not possess. All we know, of course, is the sampling $X \subset F$. However, if local knowledge of F was available, this might prove sufficient to construct $SC(X)$. Now, F is locally approximated at a point x by the tangent space to F at x, so, if F can be thought of as a collection of tangent spaces, then perhaps some progress can be made.

In fact, this is exactly what is done. In Step 1, the approximating tangent space is calculated at each point. In Step 2, this information about the local behavior of the manifold is used to find the edge-set in a manner which is exactly parallel to intersecting the Voronoi Diagram of the points with this local approximation of the manifold. However, the advantage of doing things in the way described in the previous section is in terms of the complexity, as discussed in the next section.

4.2 Complexity

To analyze the complexity of the overall algorithm, it will be simplest to break it down step by step. Complexity is in terms of N, the number of samples. (Terms which are purely in K, the dimension of the manifold, and which are independent of N are dropped.)

Finding the Neighborhoods. For a given x, this step is $O(rKN) = O(N)$ since the neighborhood is of size rK. (In fact, the step could be $O(N \log N)$: All of the points could be sorted in terms of their distance with respect to a fixed point; however, we will assume that $rK < \log N$.) Thus, for all N points, this step is $O(N^2)$.

Step 1. In the worst-case scenario, all combinations of $K+1$ points out of the neighborhood, which is of size rK, must be searched. As a result, for a single point, the complexity is

$$\binom{rK}{K+1} < (rK)^{K+1} = O(1);$$

for all points, it is therefore $O(N)$.

Step 2. As was mentioned in Section 3, the calculation of the local region amounts to a convex hull calculation of N points in K dimensions; this has complexity $O(N^{\lceil K/2 \rceil})$. This must be perfomed for for each sample, leading to a complexity of $O(N^{\lceil K/2 \rceil + 1})$.

Step 3. In principle, an edge-set $ES(x)$ may have as many as N (or really $N-1$ elements); in practice, of course, this is highly unlikely. However, assuming that this is the case, then the complexity of dealing with a single edge-set is $\binom{N}{K} < N^K$; so, the complexity of constructing the simplicial complex from all of N edge-sets is $O(N^{K+1})$.

If N is large, the complexity of Step 3 dominates so that the complexity of the entire algorithm is $O(N^{K+1})$. However, in a typical problem, $ES(x)$ should be $O(1)$ for all $x \in X$; as a result, the complexity from Step 2 dominates and the complexity is $O(N^{\lceil K/2 \rceil + 1})$. In fact, however, we may lower the complexity further if we are willing to sacrifice a little bit of certainty. The algorithm still works reasonably well if the convex hull calculation uses only the rK nearest neighbors, rather than all N points. In that case, the complexity from Step 2 should be $O(N(rK)^{\lceil K/2 \rceil}) = O(N)$; this then implies that the overall complexity will be dominated by the process of finding the neighborhoods, and is thus $O(N^2)$. (Of course, despite Steps 1-3 having a total complexity of $O(N)$, the prefactor multiplying N, which is ignored by the $O(\cdot)$ notation, may be quite considerable.)

5 Examples and Conclusions

In order to demonstrate the efficacy of the reconstruction algorithm, several examples are shown in Figs. 3, 4, 5, 6, 7, and 8. Owing to the nature of what can be visualized, only three types of examples are reproduced: Here one-manifolds (curves or disconnected curves) embedded in $\mathbb{R}^2$, one-manifolds embedded in $\mathbb{R}^3$, and two-manifolds (surfaces or disconnected surfaces) embedded in $\mathbb{R}^3$. Of course, in cases of interest, we are often looking for, say, three-manifolds embedded in $\mathbb{R}^{40}$; however, there is no good way of visualizing this. In all cases, $r = 10$ is used.

In each case, the samples are shown along with the reconstructed manifold; despite the varying geometries and topologies, the algorithm is correct in nearly all of its reconstructions. It should be noted, however, that there are cases in which the algorithm fails; see Fig. 9. The failure is

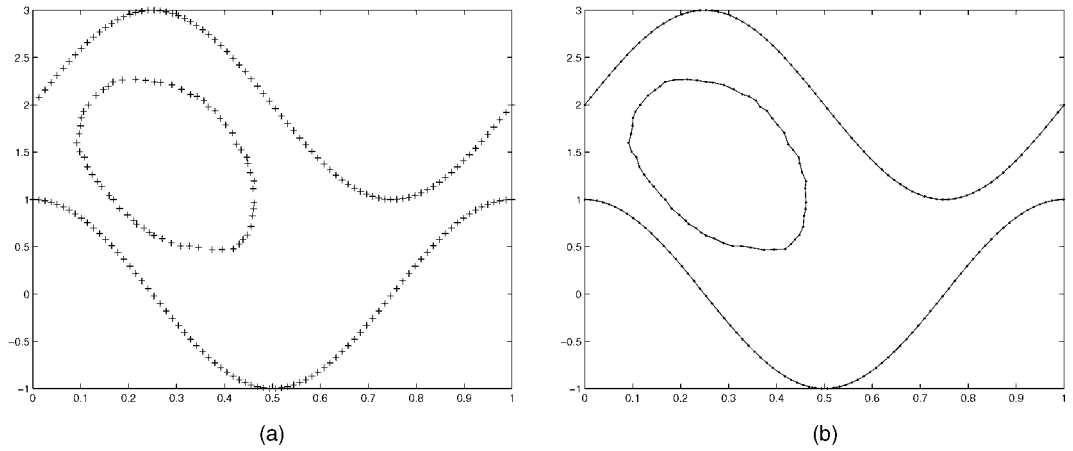

Fig. 4. A disconnected curve in the plane. (a) The samples. (b) The reconstructed manifold.

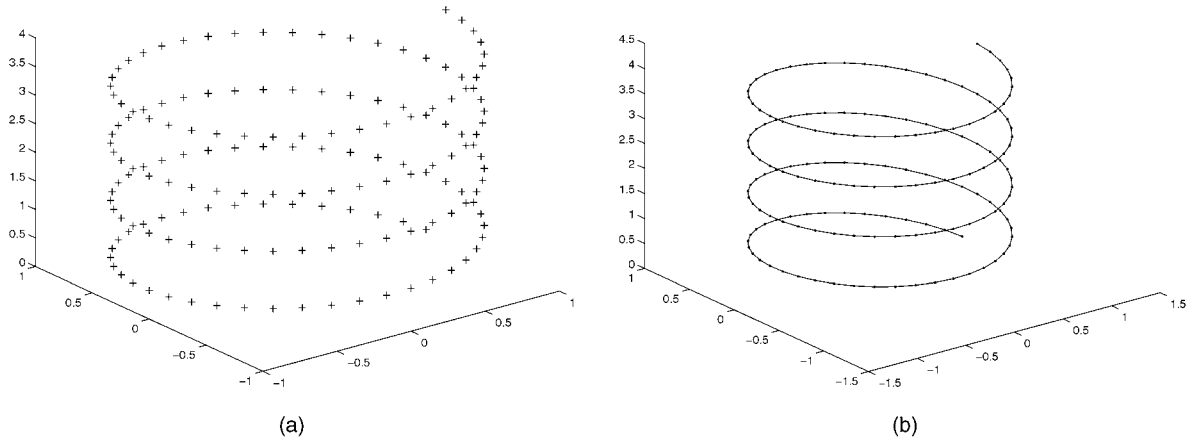

Fig. 5. A spiral. (a) The samples. (b) The reconstructed manifold.

caused by the presence of cyclic quadrilaterals, which is a general curse for Delaunay type methods: These methods cannot decide which diagonal to include. More specifically, there are two types of problems which arise. First, there are the squares, where both diagonals appear, so that the square is effectively rendered as a degenerate tetrahedron. Second, there are squares with no diagonals; this type of error is seen only once in Fig. 9. In the first case, the manifold constructed is a little "thicker" than it should be, i.e., there is a spurious tetrahedron present, like a callous. In the second case, the reconstructed manifold contains a small hole, which doesn't exist in the underlying manifold. Both errors are significant in their own ways; it may be that holes are slightly worse than callouses because they alter the homotopy type of the space.

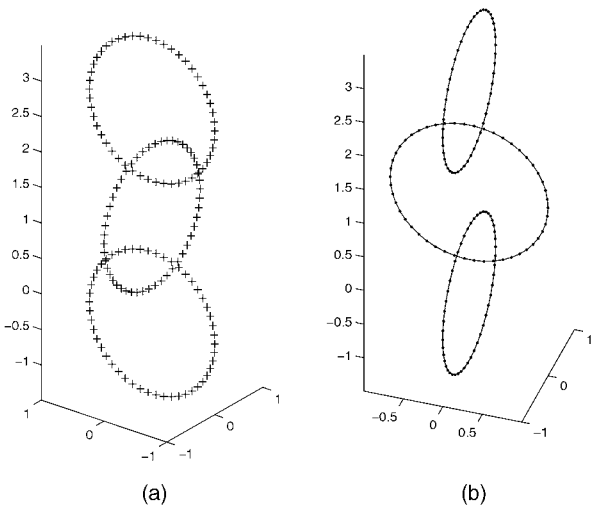

Fig. 6. Interlocking circles. (a) The samples. (b) The reconstructed manifold.

The following are the major directions for future research. First, the algorithm must be improved to avoid the failures mentioned above. In particular, both callouses and holes must be removed from the reconstructed manifolds. Second, it would be desirable to theoretically establish some bounds on sampling density under which the algorithm will be expected to succeed. This sort of result would take the form "for all sampling densities smaller than ρ_0, the algorithm will generate a homeomorphic reconstruction of the underlying manifold;" the key part of the result consists of finding ρ_0. A third direction might focus on ways of decreasing the algorithm's complexity. Methods of achieving greater efficiency were suggested in an informal way at the end of Section 4.2, the goal would be to formalize these ideas. Fourth,

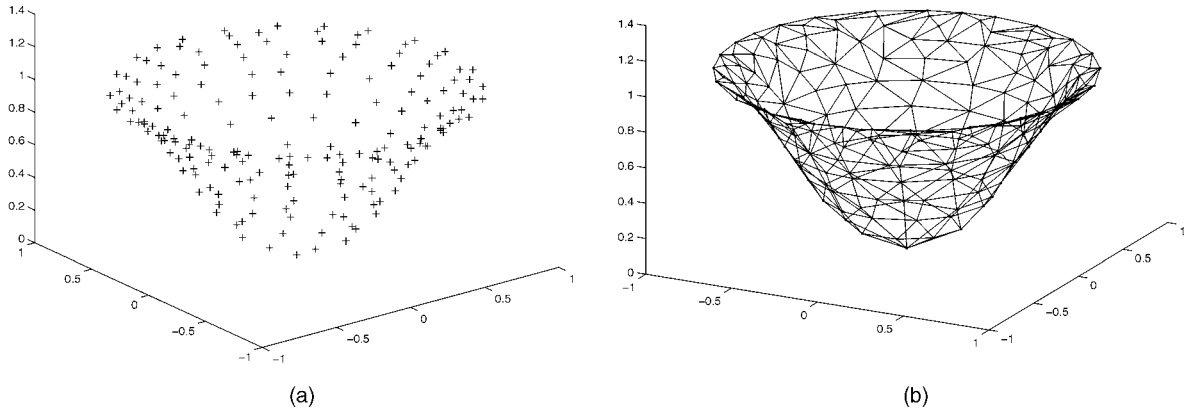

Fig. 7. The fruit bowl. (a) The samples. (b) The reconstructed manifold.

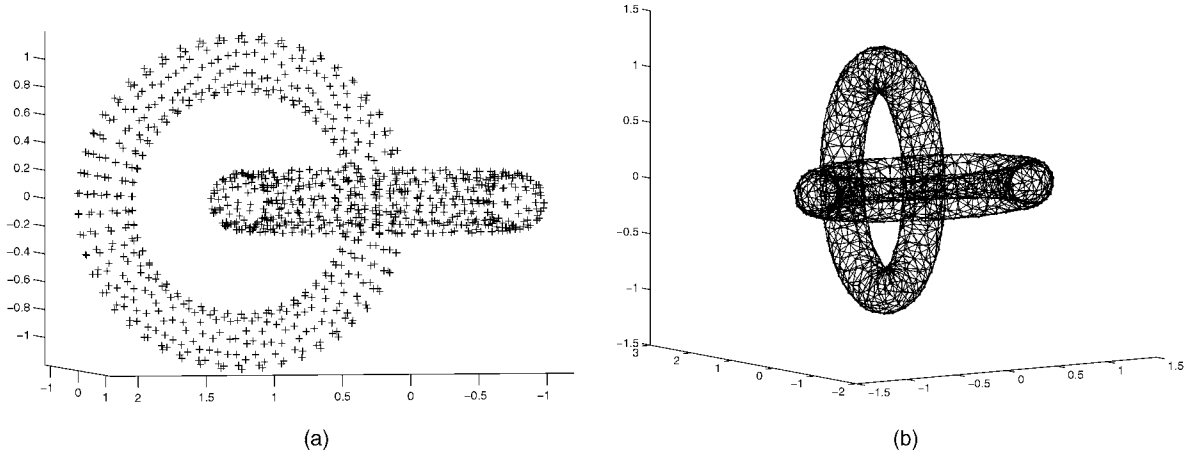

Fig. 8. Interlocking tori. (a) The samples. (b) The reconstructed manifold.

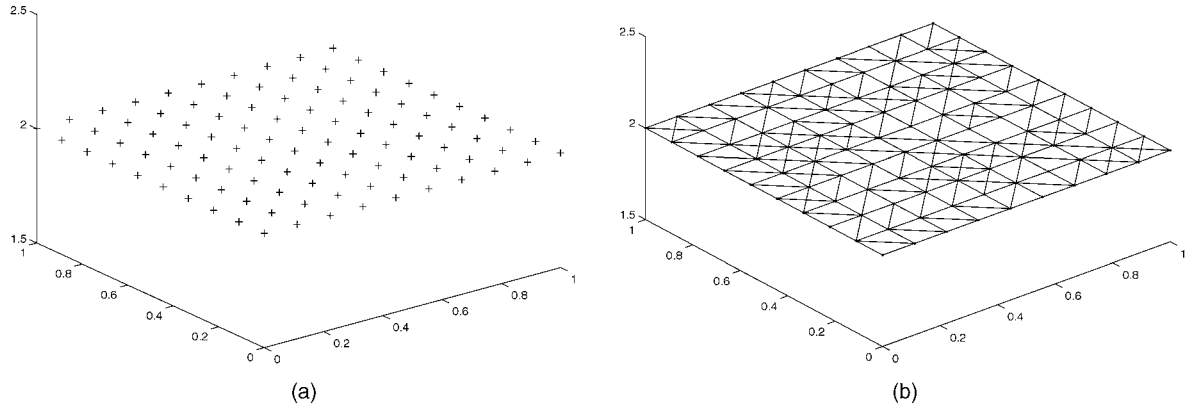

Fig. 9. Failure of the algorithm. (a) The samples. (b) The reconstructed manifold.

the algorithm must be able to address the problem of noise, particularly noise which is normal to the manifold. Certainly, such noise will destabilize the ATS calculations; the critical issue is how to circumvent this problem. Finally, given the latter improvement, it would be natural to try out this algorithm on real, physically meaningful data sets.

REFERENCES

[1] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active Contour Models," *Proc. First IEEE Int'l Conf. Computer Vision,* June 1987.

[2] A. Blake and M. Isard, "Condensation-Conditional Density Propagation for Visual Tracking," *Int'l J. Computer Vision,* vol. 29, no. 1, pp. 5-28, 1998.

[3] M. Isard and A. Blake, "Icondensation: Unifying Low-Level and High-Level Tracking in a Stochastic Framework," *Proc. Fifth European Conf. Computer Vision,* pp. 893-908, 1998.

[4] D. Freedman and M. Brandstein, "Provably Fast Algorithms for Contour Tracking," *Proc. 2000 IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 139-144, 2000.

[5] N. Amenta and M. Bern, "Surface Reconstruction by Voronoi Filtering," *Discrete and Computational Geometry,* vol. 22, pp. 481-504, 1999.

[6] N. Amenta, M. Bern, and D. Eppstein, "The Crust and the Beta-Skeleton: Combinatorial Curve Reconstruction," *Graphical Models and Image Processing,* vol. 60, no. 2, pp. 125-135, 1998.

[7] S. Haykin, *Adaptive Filter Theory,* third ed. Upper Saddle River, N.J.: Prentice Hall, 1996.
[8] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface Reconstructions from Unorganized Points," *Proc. SIGGRAPH '92,* pp. 71-78, 1992.
[9] H. Hoppe, T. DeRose, T. Duchamp, H. Jin, J. McDonald, and W. Stuetzle, "Piecewise Smooth Surface Reconstruction," *Proc. SIGGRAPH '94,* pp. 19-26, 1994.
[10] B. Curless and M. Levoy, "A Volumetric Method for Building Complex Models from Range Images," *Proc. SIGGRAPH '96,* pp. 303-312, 1996.
[11] C. Bregler and S. Omohundro, "Nonlinear Manifold Learning for Visual Speech Recognition," *Proc. Sixth IEEE Int'l Conf. Computer Vision,* pp. 494-499, 1995.
[12] N. Amenta, M. Bern, and M. Kamvysselis, "A New Voronoi-Based Surface Reconstruction Algorithm," *Proc. SIGGRAPH '98,* pp. 415-421, 1998.
[13] N. Amenta and S. Choi, "One-Pass Delaunay Filtering for Homeomorphic 3D Surface reconstruction," Technical Report TR99-08, Univ. of Texas at Austin, 1999.
[14] T. Dey, K. Melhorn, and E. Ramos, "Curve Reconstruction: Connecting Dots with Good Reason," *Proc. 15th Symp. Computational Geometry,* pp. 197-206, 1999.
[15] T. Dey and P. Kumar, "A Simple Provable Algorithm for Curve Reconstruction," *Proc. ACM-SIAM Symp. Discrete Algorithms '99,* pp. 893-894, 1999.
[16] S. Cheng and T. Dey, "Improved Construction of Delaunay Based Contour Surfaces," *Proc. ACM Symp. Solid Modeling and Applications,* pp. 322-323, 1999.
[17] E. Althaus and K. Melhorn, "TSP-Based Curve Reconstruction in Polynomial Time," *Proc. ACM-SIAM Symp. Discrete Algorithms '00,* pp. 686-695, 2000.
[18] T. Dey and R. Wenger, "Reconstructing Curves with Sharp Corners," *Proc. 16th Symp. Computational Geometry,* pp. 233-241, 2000.
[19] T. Dey and J. Giesen, "Detecting Undersampling in Surface Reconstruction," *Proc. 17th Symp. Computational Geometry,* pp. 257-263, 2001.
[20] C. Gold and J. Snoeyink, "A One Step Crust and Skeleton Extraction Algorithm," *Algorithmica,* vol. 30, no. 2, pp. 144-163, 2001.
[21] K. Mulmuley, *Computational Geometry: An Introduction through Randomized Algorithms.* Englewood Cliffs, N.J.: Prentice Hall, 1994.

Daniel Freedman (M'00) received the AB degree in physics from Princeton University in 1993 and the PhD degree in engineering sciences from Harvard University in 2000. He has been at Rensselaer Polytechnic Institute in Troy, New York, since 2000, where he is currently an assistant professor in the Department of Computer Science. He is a member of Sigma Xi, the IEEE, and IEEE Computer Society. His research interests include computational geometry, computer vision, and image processing.