A Bayesian Approach to Manifold Topology Reconstruction

Art Tevs, Michael Wand,
Ivo Ihrke, Hans-Peter Seidel

MPI-I-2009-4-002 January 2010

Authors' Address

Art Tevs, Michael Wand, Ivo Ihrke, Hans-Peter Seidel
Max-Planck-Institute für Informatik
Stuhlsatzenhausweg 85
66123 Saarbrücken
Germany

Acknowledgment

We would like to thank Martin Bokeloh (WSI/GRIS, Uni Tübingen), Ruxandra Lasowski (MPI Informatik, Saarbrücken) and Nils Hasler (MPI Informatik Saarbrücken) for their helpful and constructive comments on our paper.

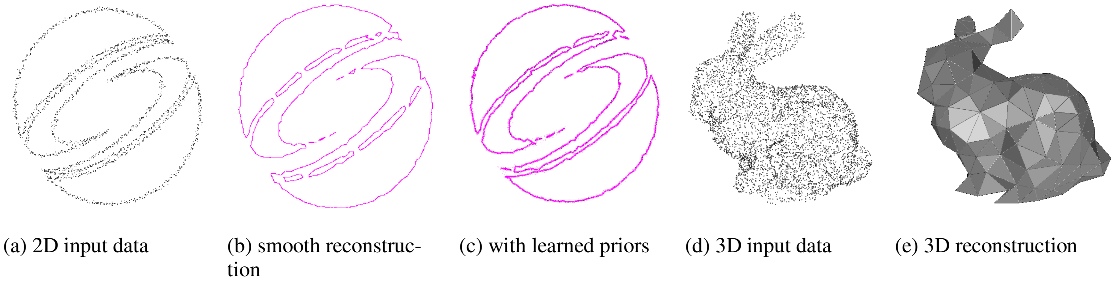
Abstract

In this paper, we investigate the problem of statistical reconstruction of piecewise linear manifold topology. Given a noisy, probably undersampled point cloud from a one- or two-manifold, the algorithm reconstructs an approximated most likely mesh in a Bayesian sense from which the sample might have been taken. We incorporate statistical priors on the object geometry to improve the reconstruction quality if additional knowledge about the class of original shapes is available. The priors can be formulated analytically or learned from example geometry with known manifold tessellation. The statistical objective function is approximated by a linear programming / integer programming problem, for which a globally optimal solution is found. We apply the algorithm to a set of 2D and 3D reconstruction examples, demonstrating that a statistics-based manifold reconstruction is feasible, and still yields plausible results in situations where sampling conditions are violated.

Keywords

Surface reconstruction, topology reconstruction, machine learning and graphics, linear programming

Contents


(a) 2D input data

(b) smooth reconstruction

(c) with learned priors

(d) 3D input data

(e) 3D reconstruction

Figure 1: *This paper discusses a new technique that extracts the most likely connectivity structure of one and two dimensional manifolds. (a) Improperly sampled 2D input data; within large parts of the object, the noise level is comparable to the local feature size. (b) Reconstruction with simple smoothness assumptions. (c) Using prior knowledge about the shape geometry, a mostly correct reconstruction is obtained. (d) Noisy sample of the Stanford bunny data set (e) Topologically correct 3D reconstruction.*

1. Introduction

Reconstruction of surfaces from point clouds is a key problem in computer graphics: The canonical area of application is reconstructing manifold meshes from 3D scanner data. The 3D scanning device yields a set of unorganized points and the task of the reconstruction algorithm is to retrieve a valid triangle mesh representation of the acquired object. Reconstruction from point clouds arises in other application areas as well: For example, the problem of extracting a low dimensional, probably curved, manifold from which a set of high dimensional data points, corrupted by noise, are originating is a classic machine learning problem. In computer graphics, such non-linear manifold learning techniques have been used for example for finding low dimensional parametrizations of BRDF [MPB*03] or articulated motion [GMH*04] from input data samples.

From a formal perspective, a surface reconstruction algorithm gets a set of point samples as input that have been taken from an unknown smooth manifold. Exploiting additional knowledge about the sampling process and expectations on the shape of the original manifold, the reconstruction algorithm's task is to reconstruct the original manifold as closely as possible. Typically, the output is a triangle mesh that has to obey to topological consistency conditions (no T-junctions, 2-manifold triangulation). Conceptually, the problem can be decomposed into two separate tasks: First, the reconstruction algorithm has to determine the *topology*[1] of the surface. This means a parametrization of the surface (typically a triangle mesh) has to be found that is homeomorphic to the original surface. Second, the reconstruction algorithm has to reconstruct the *geometry*, i.e., find the optimal embedding of the computed topology in R^n. In practice, this usually means that a base mesh that is topologically equivalent to the original surface has to be refined and smoothed in order to obtain a final surface that is close to the data points but does not contain noise artifacts from the scanning process. Given a mesh, this can be formulated as a regularized least-squares fitting problem [DTB06] , [JWB*06].

A large number of methods for surface reconstruction have been proposed. These methods can be roughly divided into two classes: Level set methods reconstruct geometry and topology in one step by fitting an implicit function to the data points. These methods typically do not give guarantees on the validity of the

[1] We use the term topology in the sense of local surface connectivity structure. This is a more detailed piece of information than the mathematical global topology (characterized only by a few numbers such as genus and number of connected components).

reconstructed topology [HDD*92], [TB99], [CBC*01], [OBA*03]. The second class is the class of Delaunay triangulation based techniques that reconstruct a mesh that is homeomorphic to the original surface if certain sampling conditions are satisfied [ABK98], [ACK01]. Usually, this means that the spacing between sample points must be at most a fraction of the local feature size, which is the distance from points to the medial axis of the original surface. Mesh filtering techniques can be applied subsequently in order to remove noise artifacts [TAU95], [DMS*99].

Our paper addresses the problem of topology reconstruction. Our perspective, however, is different from earlier work: Our goal is the reconstruction of the most likely surface in a statistical sense. This means, the task is to find the reconstruction that has a maximum a posteriori probability density for representing the original surface well, given the input data points and prior assumptions on the class of expected original surfaces. Computing the most likely reconstruction often resembles real world reconstruction tasks more closely than reconstruction under classic sampling conditions: In many real-world applications, the input point clouds have been acquired by some process (3D scanning, data collection) for which it is very hard to enforce strict sampling conditions. Therefore, we cannot hope for a correct reconstruction, but we still might aim at finding the most likely reconstruction.

Statistical techniques in 3D surface reconstruction have recently gained some interest [DTB06], [JWB*06]. However, these previous techniques all consider only the problem of smoothing a surface in a statistically optimal sense. They explicitly or implicitly require a connectivity graph or mesh as input and only perform statistically optimal smoothing in a Bayesian maximum a posteriori sense. In case no mesh is given, these techniques cannot be applied [DTB06] or create artifacts in case of undersampled input point clouds [JWB*06].

Our paper tries to complement these techniques by a Bayesian formulation of topology reconstruction: We first derive a generative statistical model of sampling points from meshes and use Bayes' rule to derive an inference problem. We then approximate this model to obtain a computationally tractable formulation, which is solved by an integer programming based optimization algorithm. We apply an implementation of this algorithm to several 2D and 3D data sets. The a priori model of the reconstructed surfaces can be learned from example data sets; alternatively, a simpler general analytical smoothness model can be employed.

The resulting optimization problems are computationally intense. Therefore, our current implementation is limited to handling rather small data sets. In terms of computation times, the technique is not yet competitive to state of the art 3D reconstruction algorithms. We see the main contributions of this work in for the first time examining a statistical formulation of the topology reconstruction problem. In particular, this allows for incorporating explicit models of the acquisition process as well as prior knowledge about the class of objects in an easy and general way. It turns out that the incorporation of such prior knowledge can significantly improve the reconstruction quality.

2. Related Work

Surface reconstruction is a classic problem in computer graphics. However, only a few statistically motivated approaches exist that are able to take prior knowledge into account. In this section we give a brief, non-exhaustive overview of general surface reconstruction techniques and focus in particular on robust and statistical techniques.

Implicit functions: Based on pioneering work of Hoppe et al [HDD*92], a large number of approaches employ implicit function fitting for surface reconstruction. Turk and O'Brien [TB99] use globally supported radial basis functions for fitting. The complexity can be reduced using a fast multipole method [CBC*01]. Ohtake et al. [OBA*03] define the surface locally via quadratic functions that are blended using partition of unity weights. Moving Least Squares (MLS) approaches are a special case of implicit function-based-techniques. They define a surface as an invariant set of a projection operator. The projection operator is defined as a numerical optimization step on a locally constructed implicit function [ABC*03]. Implicit function based reconstruction techniques typically do not give guarantees for topological correctness of the solutions. Kolluri R. [KOL05] proves topological correctness and bounded approximation error for an MLS scheme under sampling conditions similar to those required by Voronoi based reconstruction algorithms (see below).

Mesh Smoothing: A number of filtering techniques have been proposed to remove noise from triangle meshes, in analogy to image filtering operations [TAU95], [DMS*99], [DMS*00]. These techniques and their follow up work require a topologically correct mesh as input.

Voronoi- / Delaunay-based techniques: Amenta and colleagues approached the reconstruction problem from a computational geometry point of view [ABK98], [ACK01]. Under sufficient sampling conditions (sample spacing smaller than a fraction of the distance to the medial axis of the original shape), the restricted Delaunay triangulation is homeomorphic to the original surface, and it can be computed by labeling poles in the 3D Delaunay tetrahedrization. The original algorithms are quite sensitive to noise. This problem has been addressed in [MAV*05], which extends the power crust algorithm to noisy samplings with a noise level not exceeding a fraction of the local feature size. Another extension towards robustness is proposed in [KSB04]: After labeling Voronoi cells and poles, a global graph cut optimization algorithm is employed to extract a globally optimal surface. Their approach is efficient and robust to noise and outliers. Alliez et al. [ACT*07] propose an algorithm employing a mixture of implicit function fitting and Voronoi tessellation that solves a general eigenvalue problem to avoid fixing normal orientations locally. However, the two latter techniques are not able to take into account specific prior knowledge. In addition, they are based on a cell decomposition of space, and thus, in contrast to our algorithm, hard to generalize to the case of low dimensional manifolds embedded in higher dimensional spaces ($d > 3$). The idea of enforcing topological consistent triangulations using linear programming was introduced in [AM00], [AGJ00] and [AGJ02]. We extend this idea to a global, statistically-based optimization method that permits using geometric shape priors.

Statistical learning: Only a few previous surface reconstruction techniques have been based on a statistical framework. Steinke et al. [SSB05] use support vector machines for reconstruction, hole filling and morphing between datasets. Pauly et al. [PMG04] quantify uncertainty in point cloud data by looking at locally interpolated surface pieces. As highlighted in the introduction, Diebel et al. [DTB06] describe a Bayesian mesh filtering technique, and Jet al. [JWB*06] use a Bayesian approach to point cloud filtering. An improved, faster and adaptive version is described by Huang et al. [HAW07]. Both algorithms do not attempt an explicit topology reconstruction; in this sense our technique is complementary to their approaches. Using example shapes to define prior knowledge has been proposed in [GSH07] and [DTB06]. Gal et al. use shape matching with example shapes to improve MLS surface fitting results and Diebel et al. learn distributions of angles in meshes, similar to our prior model. Neither of the two approaches aims at topology reconstruction. A technique for statistical topology reconstruction has been proposed by Aupetit [AUP05]: It maximizes the likelihood of a set of segments generating data point samples using an EM algorithm. Results are reported on very low complexity 2D reconstructions. The technique does not use any geometric priors in the reconstruction. Therefore, it probably does not generalize to more complex reconstruction tasks. Niyogi et al. [NSW04] provides a prove and precise conditions under which one can infer the correct topology of an embedded manifold from a noisy sample. The authors also include an estimate of the probability of the reconstruction being correct.

3. Statistical Model

We formulate our reconstruction as Bayesian inference problem: We model the creation of data sets as a two step random experiment. First, the original scene S is chosen from a probability space Ω_S according to a prior distribution $p(S)$ that models our expectations of the objects we have to deal with. Then, in a dependent random experiment, a data set of sample points D from S is determined, governed by a measurement distribution (likelihood) $p(S|D)$ defined on a probability space $\Omega_S \times \Omega_D$, where Ω_D comprises the set of all possible point clouds generated by the data acquisition process.

3.1 Probability Spaces

First, we need to define the probability space we are operating on. In this paper, we will consider both the reconstruction of 1-manifolds (curves), from 2D point samples and the reconstruction of 2-manifolds (surfaces), from 3D point samples. In the 2D case, we assume that curves are piecewise linear, that is, they consist of a finite set of straight line segments. In 3D, we model piecewise linear objects by triangle meshes. According to our Bayesian reconstruction approach, we assume that the original surfaces have been of this type, i.e. have been piecewise linear. In addition, we also assume that we know an upper limit for the complexity; no input will have more than a fixed number n_{max} of segments / faces. This is no restriction in practice but makes the formal formulation easier.

This leads to the following definition of the probability space of original scenes: Ω_S is the set of all sets of line segments (2D) / triangles (3D) of size no larger than n_{max} and which are *topologically consistent*. A set of line segments is considered to be topologically consistent, if the following conditions hold:

2D topological consistency: The line segments form a set of closed[2] curves, i.e., every vertex is incident to exactly two line segments. In addition, the curve is a 1-manifold: No line segments are allowed to intersect. For the 3D case, the conditions are very similar:

3D topological consistency: We again demand for a closed 2-manifold, i.e., every edge of a triangle must be incident to exactly one other triangle; intersections in the interior of triangles as well as non-manifold conditions at vertices (more than one surface fan) or edges (more than two adjacent triangles) are not allowed.

This means, we will a priori exclude topologically invalid objects from our probability space so that they cannot be the result of the reconstruction.

[2] For simplicity, we consider closed curves and surfaces only in this paper. A generalization to manifolds with boundary is mostly straightforward.

3.2 Likelihood Model

Next, we define the measurement model. Given we know that a piecewise linear curve (2D) or triangle mesh (3D) has been the original model, we need to evaluate the likelihood that a set of data points has emerged from this original surface. Similar to [WJH*07], we assume that the data acquisition process samples the input manifold independently for each point according to a sampling distribution $p_{sampl}(x)$, $x \in S$. Each sample is then independently distorted by Gaussian noise according to a normal distribution function $N_{\mu(x),\sigma(x)}$. The likelihood of a single sample point given the original surface S is therefore given by:

$$p(\mathbf{d}_i \mid S) = \int_S p_{sampl}(\mathbf{x}) \cdot N_{\mu(\mathbf{x}),\sigma(\mathbf{x})}(\mathbf{x} - \mathbf{d}_i) d\mathbf{x}, \qquad (1)$$

For simplicity, we assume uniform sampling and unbiased, identically distributed Gaussian noise with an isotropic covariance matrix characterized by a single global standard deviationσ. Equation (1) then simplifies to:

$$p(\mathbf{d}_i \mid S) = \int_S N_\sigma(\mathbf{x} - \mathbf{d}_i) d\mathbf{x} \qquad (2)$$

As each sampling point is created independently, the overall likelihood of n data points $\mathbf{d}_i$ is given by:

$$p(D \mid S) = \prod_{i=1}^{n} p(\mathbf{d}_i \mid S) \qquad (3)$$

In negative log space, this corresponds to the potential function

$$E(D \mid S) = \sum_{i=1}^{n} \log \int_S N_\sigma(\mathbf{x} - \mathbf{d}_i) d\mathbf{x}. \qquad (4)$$

Let s_i be the i-th out of n_S segments (lines segments, triangles) the original scene S consists of. We then obtain the final neg-log-likelihood as:

$$E(D \mid S) = \sum_{i=1}^{n} \log \sum_{j=1}^{n_S} \int_{s_i} N_\sigma(\mathbf{x} - \mathbf{d}_i) d\mathbf{x}. \qquad (5)$$

3.3 Prior Model

Any reconstruction problem is only well defined, if additional assumptions on the original input model are made, such as smoothness or other types of simplicity assumptions. The topological constraints imposed so far are not yet sufficient in order to constrain the geometric embedding of the surface. In particular, a space filling surface densely covering the area adjacent to the data points might obtain a higher score than a simple smooth surface, as more of the surface area would be concentrated close to the data points. Within a Bayesian inference framework, prior probabilities on the original models (in our case, the distribution P(S) given on Ω_S) are used to regularize the problem. In this paper, we consider two types of priors – simple smoothness assumptions and shape priors learned from example data [DTB06]. Both types of priors can be expressed with the same formalism:

2D shape priors: In the 2D case of reconstructing piecewise linear, 1-manifold curves, we formulate our priors in terms of distributions of discretized curvature. We assume that we are given a probability distribution $p_{curv}(\boldsymbol{\kappa})$ on unsigned curvature values $\boldsymbol{\kappa}$. We then formulate the prior probability as a Markov random field: For each pair of adjacent line segments in our solution S, we compute the curvature of a simple interpolating curve[3] at the center point and assign a clique potential proportional to this distribution function to each pair. A modification of $p_{curv}(\boldsymbol{\kappa})$ will allow us to specify different expectations on the space of original models. For example, an exponential distribution $\mu e^{-\mu\boldsymbol{\kappa}}$ will prefer smooth objects, with smoothness controlled by the parameter μ. A mixture with a Gaussian model with maxima around discrete angles such as 90° and 180° will prefer objects with right angle corners.

3D shape priors: In the 3D case, we follow the same concept. We compute a curvature value according to the angle at which two adjacent triangles meet over their common edge. By controlling this distribution function, we can describe different classes of original objects.

In both cases, we obtain a 2-clique Markov random field for the prior distribution, given as:

$$p(S) \;=\; \prod_{s_i, s_j \text{ adjacent in } S} p_{curv}\big(\boldsymbol{\kappa}(s_i, s_j)\big) \tag{6}$$

In neg-log space, this corresponds to a potential of the form

$$E(S) \;=\; \sum_{s_i, s_j \text{ adjacent in } S} e_{i,j}^{(curv)}, \tag{7}$$

where $e^{(curv)}$ encodes pairwise penalties for having adjacent segments in the solution. If the curvature across these segments meets our prior expectations, the penalty will be smaller than otherwise.

3.4 Inverse Problem

Having defined the likelihood and prior models, the probability function to be optimized by a best reconstruction S is given by Bayes rule:

$$p(S \mid D) = \frac{p(D \mid S)\,p(S)}{p(D)} \tag{8}$$

As $p(D)$ is constant during optimization, it is sufficient to minimize the sum of the two remaining neg-log-probabilities:

$$\text{compute: } \arg\min_{S} E(S \mid D) = E(D \mid S) + E(S) \tag{9}$$

[3] In the actual implementation, we do not need to perform this step explicitly; therefore, we do not specify an implementation here. A canonical choice could be interpolating quadratic spline curves.

4. Discretization and Optimization

So far, we have developed a statistical model that assigns a probability density to every triangle mesh of bounded complexity that approximates the data. The remaining problem is to devise an algorithm that computes the solution that maximizes the a posteriori probability density. Unfortunately, a direct optimization of the exact statistical potential functions is a very difficult problem. Therefore, we have to replace our strict model with an approximation that can be handled computationally.

4.1 Discrete and Continuous Optimization

We have a continuous and a discrete component in our model, which have to be optimized simultaneously. The discrete component is the topological graph of segments (i.e., the graph of line segments or the triangle mesh). We have to determine this graph, choosing only from topologically valid solutions, in a way such that a good geometric embedding can be obtained. The continuous component is the placement of the vertices; given a topology, we need to place the segments in space so as to maximize the likelihood of the data. The first simplification we perform is to remove the continuous problem and replace it by a fully discrete optimization problem.

For this purpose, we define a set $\boldsymbol{K}$ of *keypoints* that are to be connected with segments (two keypoints with a line segments in 2D, three keypoints with a triangle in 3D). The keypoints are obtained by subsampling the input data points with a Poisson disc sampler. The Poisson disc sampler tries (in a greedy manner) to keep as many of the original data points as possible while maintaining a minimum distance ε_{key} between all data points. The sample spacing ε_{key} is chosen above the noise level in the data (the noise level σ is either known in advance, due to scanner characteristics, or it is set empirically) so that angles between connecting segments are meaningful and can be used to evaluate our priors. During the optimization, we will fix the location of these keypoints, thus removing the continuous optimization component. In a postprocessing step, we can then use a Bayesian smoother (such as [DTB06], [JWB*06]) to optimize the geometric embedding[4]. In order to improve the quality of keypoint placement, we perform an additional, heuristic optimization. A typical data set in a reconstruction set will consist of "good" and "bad" parts; in the good parts, the geometry is simple (e.g. locally flat, no nearby parts) and topology reconstruction is easy, while bad parts are more challenging. We try to estimate good parts by performing a principal component analysis in a Euclidian neighborhood of $c\sigma$ (typically $c \approx 2.5$ which is chosen empirically) of the original data. The ratio between the smallest and the largest eigenvalue estimates how well a Euclidian neighborhood covers the original surface patch. We use the ratio to smoothly blend between the computed average of the neighborhood and the original position of the keypoint. This behavior pre-smoothes the results in areas where the

[4] It might appear natural to iterate this process; however, fixing the topology is so decisive for the result of the smoothing step that an iterated topology computation would not improve the result further [Jenke et al. 2006].

normal deviation is small and does not change anything where the situation is ambiguous. As this step is done during Poisson disc sampling, it automatically generates a smooth and less dense sampling of areas that appear simple to our heuristic and produce more keypoints (distributed over a volumetric area of noisy points in the 3D case) if no pre-smoothing can be performed. In practice, this heuristic reduces the sampling requirements considerably and leads to better reconstruction results.

Having defined a set of keypoints, we next compute a set of candidate segments our algorithm has to choose from. For this purpose, in the 2D case, we connect all pairs of points within distance of $c'\varepsilon_{key}$ with line segments that are potentially included in the solution (typically: $c' \approx 3$, hence enclosing at least 2 neighbor points of Poisson-sampled keypoints). In the 3D case, we create all triangles of three points that can be included in a ball of radius $c'\varepsilon_{key}$.

We now obtain a purely discrete optimization problem: The variables to be optimized are the segments $s_i \in \{0,1\}, j = 1\ldots k$, where $s_i = 1$ refers to including and $s_i = 0$ to excluding the segment s_i from the solution.

4.2 Approximate Likelihood

The next problem we are facing is that our likelihood model is non-local: Due to the logarithm in Equation (5), the increment in penalty for a single segment depends on the set of already chosen segments. If the data points are already well covered with segments, the increment is smaller than in the case where some data point that was previously uncovered obtains a nearby segment for the first time. In order to obtain a local, linear model, we employ an approximate likelihood that computes the segment matching score for each segment separately:

$$\begin{aligned}\widetilde{E}(D \mid S) &= \sum_{i=1}^{n}\sum_{j=1}^{n_S} match(d_i, s_j)\\ &= \sum_{j=1}^{n_S}\sum_{i=1}^{n} match(d_i, s_j) \qquad (10)\\ &= \sum_{j=1}^{n_S} e_j^{(data)}.\end{aligned}$$

The matching function $match(d_i, s_j)$ is computed as a normalized sum over all reciprocal distances of data points d_i which lie in the radius σ around the segment s_j. Hence only the points which are approximated by the segment do contribute to the matching score. We approximate the integral over the Gaussian noise in Equation (1) by a simple quadratic point-to-segment distance. Our approximate likelihood has an artifact we need to remove: We can obtain an optimal score by using no segments at all. In order to discourage such zero solutions, we assign negative costs to edges by subtracting a fixed constant from the energy associated with each edge. This constant is set to the maximum of all data costs that can occur.

Effectively, Equation (10) moves the logarithm of Equation (5) inside the sum, thus allowing us to express the matching term as a sum of penalties $e_j^{(data)}$ that occur for selecting each individual segment, independent of the others. The adverse effect of this is that covering an area of dense data points with two nearby edges still alters the likelihood, which is not statistically warranted. We can afford this type of approximation in our case because we enforce strict topological validity. In typical cases, it is not possible to create topologically correct solutions of Poisson-sampled keypoints with multiple line segments covering the same set of data points. We also drop the surface area term from Equation (5), which has only a minor effect on the solution.

4.3 Approximate Prior

The prior is a 2-clique Markov random field. This means, if we use indicator variables to turn on and off individual segments, the log-probability potential will be quadratic in these variables. For the optimization, this causes additional costs that we want to avoid. Therefore, we employ a simplified prior term that is linear in the indicator variables[5]. This is done by computing the 2-clique potentials for all pairs of potentially adjacent segments $\{s_i, s_j\}$ and computing the average of this score for the two segments s_i, s_j involved. The averaged scores are stored in a prior variable $e^{(prior)}(s_i)$. In addition to the curvature based prior, we also penalize large segments (long edges in the 2D case), as they are more “dangerous”: Large segments are more likely to connect disconnected parts of the object. Therefore, it is preferable to employ small ones as long as they are available. For the 2D case we simply penalize long edges and favor short ones. In the 3D case we use the maximal length of all triangle sides and penalize larger triangles in this sense. This additional prior could be considered as a “nearest neighbor prior”.

Learning the prior: By computing the prior scores based on the smoothness of the shape we prioritize smooth regions over the sharp ones. However in some situations it is necessary to reconstruct sharp features (see Fig.1.a). In order to achieve this, we employ a training stage enabling us to incorporate prior knowledge of the sampled data set to support the solution process. As training data we use a pre-defined segment set with correct topological and manifold connections. We use the training data to compute a histogram of the curvature. For the 2D case we sample three points which are topologically connected within a certain radius. The angle, between these segments, having one of the sampling points in common, represents one sample of the histogram. As the keypoints are placed at a rather uniform spacing by the Poisson disc sampler, the angles are a reasonable approximate measure of curvature. The histogram is then sampled to compute the 2-clique potential for the adjacent segments $\{s_i, s_j\}$. This enables us to prefer segment pairs which have higher occurrence in the training data. The additional information helps to reconstruct segments which have a higher probability to belong to the correct solution. Figures 2 and 3 show the differences between trained and non-trained priors in the solution.

4.4 Optimization

With the approximations made so far, we have introduced a simpler energy function. We have to optimize the linear function:

$$\tilde{E}(S \mid D) = \tilde{E}(D \mid S) + \tilde{E}(S) = \sum_{i=1}^{k} s_i \left(e_i^{(data)} + \lambda e_i^{(prior)} \right) \qquad (11)$$

where λ is a weighting factor to trade-off the influence of data matching and prior assumptions. For large λ the solver prefers smooth and short segments over segments that fit the data points well, while data fitting dominates for small λ. In addition to minimizing this score, we also need to enforce topologically correct solutions. We account for this requirement by introducing a set of linear constraints. As shown in [AM00], [AGJ00], and [AGJ02] topological correct tessellations can be characterized by linear inequality constraints. Each segment (line segment, triangle) is characterized by a variable s_i that is either zero or one. In the 2D case, the following linear constraints are necessary and sufficient in order to obtain topologically correct tessellations:

[5] Please note that a prior distribution in the Bayesian sense is a subjective assumption; therefore, the approximate model is not “wrong” in a technical sense. However, it models our original intention less closely.

2D Topological Validity Constraints:

Let E_i be the set indices j of all line segments s_j that are incident to one and the same vertex v_i. Let $E^{(k)}(s_i)$, $k \in \{1,2\}$, be the same set, but indexed by one adjacent segment s_i and an indicator k that chooses one of its two vertices. We require that at every vertex either no line segments meet, or exactly two, if at least one is present. This can be formulated by the following constraints:

$$\begin{gathered}\forall \text{vertices}\, v_i : \sum_{j \in E_i} s_j \leq 2 \text{ and} \\ \forall k \in \{1,2\} : \forall \text{segments}\, s_i : s_i - \sum_{\substack{j \in E(s_i), \\ j \neq i}} s_j \leq 0\end{gathered} \tag{12}$$

Next, we need to avoid intersections within segments, which can also lead to non-manifold results. Let I_i be the set of all segment indices, a given segment s_i intersects with (not counting segments that just touch at the vertices). We can then setup linear constraints as follows:

$$\forall \text{segments}\, s_i : (\# I_i) \cdot s_i + \sum_{\substack{j \in I_i, \\ j \neq i}} s_j \leq (\# I_i) \tag{13}$$

Each of these constraints avoids intersections with the corresponding segment s_i, but does not impose any constraints if the considered segment is not in the solution set.

3D Topological Validity Constraints:

In the 3D case, our segments are triangles. We can apply the same sets of constraints. For the manifold constraints (12), we now constrain the number of triangles that can meet via a shared edge. For this purpose, we just have to let E_i be the set of all triangles that meet at an edge, and $E^{(k)}(s_i)$, $k \in \{1,2,3\}$ be the sets of edge-incident triangle indices. Otherwise, the constraints are the same. The intersection constraints (13) are formulated exactly in the same way; for triangles, we do not include intersections that occur at shared vertices or shared edges only. A difference in the triangulation case is that triangles can also meet at a common vertex. We apply the same solution that is proposed in [AGJ00]: For each vertex v_i, we compute all manifold fans $F_i^{(j)}$ of triangles connected via edges that are incident to this vertex. Then, we disallow that two of them are present at the same time by requiring:

$$\begin{gathered}\forall \text{vertices}\, v_i : \forall \text{Fans } F_i^{(j_1)}, F_i^{(j_2)} \text{ adjacent to } v_i : \\ \sum_{k \in F_i^{(j_1)}} s_k + \sum_{k \in F_i^{(j_2)}} s_k \leq \# F_i^{(j_1)} + \# F_i^{(j_2)} - 1\end{gathered} \tag{14}$$

However, the number of such constraints can be large and impose a very high computational burden upon the optimization algorithm. Therefore, again following [AGJ00], we only create necessary constraints on demand: After having computed a solution, we check for non-manifold results at vertices and switch on the necessary constraints to avoid these. Then we solve again. In order to reduce the computational costs of multiple such solution passes, we first only re-solve the problem in a 1-ring neighborhood of the modified vertices, keeping the rest of the solution fixed. If this turns out to be infeasible, we try again in a 2-ring neighborhood, then a 4-ring and so on, until we finally would try to recompute a global solution, after at most O(log n_S) steps. In practice the re-solving in a 1-ring neighborhood already determines the solution with respect to the constraints.

In order to solve the resulting integer programming problem, we have tested two alternatives: First, using a continuous linear programming solver and relaxing the problem such that the variables s_i might assume any value within an interval [0,1]. This solution is fast, as solving very large linear programs is nowadays a standard problem for which very efficient algorithms are known. However, the algorithm might output indecisive solutions, such as assigning scores of 0.5 to two intersecting edges. In our experience, relaxed solutions work well for 2D examples with very strong priors (large λ). For other cases as well as 3D examples, we often obtained indecisive solutions in problematic areas, where sampling conditions are violated. The second alternative is using a full integer programming solver. This is more costly, as NP-hard classes of integer programs exist. In practice, we were able to solve problems of our kind with up to about 2000 unknowns. With discrete integer programming, no uncertainty artifacts occur and we always obtain a topologically correct solution. This technique has been used for all examples shown in this paper.

5. Implementation and Results

We have implemented our algorithm in C++, using the freely available linear programming solver SCIP [ACH04], which is capable of solving continuous and integer programming problems. The 3D results have been rendered using Geomview. Timings were obtained using an AMD Opteron 2218 CPU with 2.6GHz and 3GB RAM. Table 1 summarizes the example scenes and the timing results. Figures 2-6 show example reconstructions of the corresponding sampling data sets.

Figure 2 shows the Floor-Plan 2D data set and its topological reconstruction. The original surface contains only 90° and 180° angles between adjacent segments making it hard to compute correct topology using just smoothness assumptions. Using a training data set, consisting of only simple rectangles, the optimizer learns to prefer corners as well as flat regions, computing a solution that reconstructs the correct object topology. A similar effect can be observed in Figure 1a-c, which shows a very noisy sampling of a 2D logo image, with points offset by noise up to the medial axis of the shape in the narrow regions. Again, employing learned shape priors improves over the simple smoothness assumptions (we learn the curvature distribution of the original, high resolution 2D logo). Figure 3 shows reconstruction of boundary curves of the letters, A, B and C. Again, a priori knowledge about the segment curvature helps the optimizer to find a more suitable solution. Since the optimizer prefers smooth segment-to-segment connections the non-trained solution looks more rotund.

For the 3D case we use point clouds with different noise levels generated from the Stanford Bunny dataset. Figure 5 shows the corresponding reconstructions. Our algorithm is capable to reconstruct also very noisy datasets as shown on the right part of Figure 5. However, the estimated "most likely" topology does not match the correct topology anymore, which of course is expected. In particular, we obtain an additional connected component at the second ear of the bunny for medium noise level, and two such segments for the very high noise level (standard deviation of 5% of the scene size, i.e. we have a significant number of points up to 10% away on *both* sides of the model). The computation time increases for very noisy examples as the problem becomes less clearly determined.

Choosing parameters: We have to trade-off prior assumptions and data fitting to define the reconstruction task. Experimentally, we found that $\lambda \approx 5/2$ provides a good computation time to result quality ratio with respect to our example data sets. In addition, we had to adjust the relative weight of the segment size in the prior manually for the examples. Choosing parameters also affects the computation time: The speed of the integer programming solver depends on the optimization problem. In our case, it turns out that a large weight on curvature priors tends to increase the running time, while preferring short edges (obviously) decreases the running time. As intuitively expected, PCA-presmoothing also decreases the running time.

6. Conclusions and Future Work

In this paper, we have presented a statistically based approach to topology reconstruction, in the sense of reconstructing a topologically matching base mesh for a noisy sample of data points. The algorithm guarantees to output topologically valid, closed meshes. Using this technique, it is possible to obtain plausible solutions in the presence of strong noise artifacts. A particular feature of the Bayesian formulation is that prior knowledge can be incorporated into the optimization. This allows the algorithm to account for expected object features such as sharp angles and thus obtain significantly improved reconstructions.

Bibliography

[ACH04] ACHTERBERG, T. (2004). SCIP - A framework to integrate Constraint and Mixed Integer Programming(04-19), Technical report, Zuse Institute Berlin.

[AGJ02] ADAMY, U., GIESEN, J. & JOHN, M. (2002). Surface Reconstruction Using Umbrella Filters, *Computational Geometry - Theory and Applications* **21**, 63–86.

[AGJ00] ADAMY, U., GIESEN, J. & JOHN, M. (2000). New Techniques for Topologically Correct Surface Reconstruction, *in* T. Ertl, B. Hamann & A. Varshney, ed.,Proceedings Visualization 2000, pp. 373–380.

[ABC*03] ALEXA, M., BEHR, J., COHEN-OR, D., FLEISHMAN, S., LEVIN, D. & SILVA, C. T. (2003). Computing and Rendering Point Set Surfaces, *IEEE Transactions on Visualization and Computer Graphics* 9(1), 3–15.

[ACT*07] ALLIEZ, P., COHEN-STEINER, D., TONG, Y., DESBRUN, M. (2007): Voronoi-based Variational Reconstruction of Unoriented Point Sets. *Symposium on Geometry Processing 07*, pp. 39–48.

[AM00] ALTHAUS, E. & MEHLHORN, K. (2000). TSP-based curve reconstruction in polynomial time, *Symposium on Discrete Algorithms*, pp. 686–695.

[ABK98] AMENTA, N., BERN, M. & KAMVYSSELIS, M. (1998). A new Voronoi-based surface reconstruction algorithm, *SIGGRAPH 98*, pp. 415–421.

[ACK01] AMENTA, N., CHOI, S. & KOLLURI, R. K. (2001). The power crust, SMA 01, pp. 249–266.

[AUP05] AUPETIT, M. (2005). Learning Topology with the Generative Gaussian Graph and the EM Algorithm, *Neural Information Processing and Systems (NIPS 2005)*, pp. 83-90.

[CBC*01] CARR, J. C., BEATSON, R. K., CHERRIE, J. B., MITCHELL, T. J., FRIGHT, W. R., MCCALLUM, B. C. & EVANS, T. R. (2001): Reconstruction and representation of 3D objects with radial basis functions. *SIGGRAPH 2001*, pp. 67–76.

[DMS*00] DESBRUN, M., MEYER, M., SCHROEDER, P. & BARR, A. H. (2000). Anisotropic Feature-Preserving Denoising of Height Fields and Images. *Graphics Interface 2000*, pp. 145–152.

[DMS*99] DESBRUN, M., MEYER, M., SCHROEDER, P. & BARR, A. H. (1999). Implicit fairing of irregular meshes using diffusion and curvature flow. *SIGGRAPH 99*, pp. 317–324.

[DTB06] DIEBEL, J. R., THRUN, S. & BRÜNIG, M. (2006). A Bayesian method for probable surface reconstruction and decimation, *ACM Transactions on Graphics* **25**(1), 39–59.

[GSH07] GAL, R., SHAMIR, A., HASSNER, T., PAULY, M., COHEN-OR, D. (2007). Surface Reconstruction using Local Shape Priors. *Symposium on Geometry Processing 07.*

[GMH*04] GROCHOW, K., MARTIN, S. L., HERTZMANN, A. & POPOVIĆ, Z. (2004), Style-based inverse kinematics, *ACM Trans. Graph.* **23**(3), 522–531.

[HDD*92] HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J. & STUETZLE, W. (1992). Surface reconstruction from unorganized points, SIGGRAPH 92, pp. 71–78.

[HAW07] HUANG, Q.-X., ADAMS, B., WAND, M. (2007). Bayesian Surface Reconstruction via Iterative Scan Alignment to an Optimized Prototype. Symp. Geometry Processing 07.

[JWB*06] JENKE, P., WAND, M., BOKELOH, M., SCHILLING, A. & STRASSER, W. (2006). Bayesian Point Cloud Reconstruction, Eurographics 2006.

[KSB04] KOLLURI, R. SHEWCHUK, J.R., O'BRIEN, J.F. (2004): Spectral Surface Reconstruction from Noisy Point Clouds, *Symposium on Geometry Processing 04*, pp. 11-21.

[KOL05] KOLLURI R. (2005). Provably good moving least squares. *Proc. Symp. Discrete Algorithms 05*.

[MPB*03] MATUSIK, W., PFISTER, H., BRAND, M. & MCMILLAN, L. (2003). A Data-Driven Reflectance Model. *ACM Transactions on Graphics* **22**(3), 759–769.

[MAV*05] MEDEROS, B. AMENTA, N. VELHO, L. & DE FIGUEIREDO, L. H. (2005). Surface Reconstruction for Noisy Point Clouds. *SGP 05*, pp. 53–62.

[NSW06] NIYOGI, P., SMALE, S., WEINBERGER, S. (2006). Finding the Homology of Submanifolds with High Confidence from Random Samples, preprint, Sept. 2004

[OBA*03] OHTAKE, Y., BELYAEV, A., ALEXA, M., TURK, G. & SEIDEL, H. (2003). Multi-level partition of unity implicits, *ACM TOG* **22**(3), 463–470.

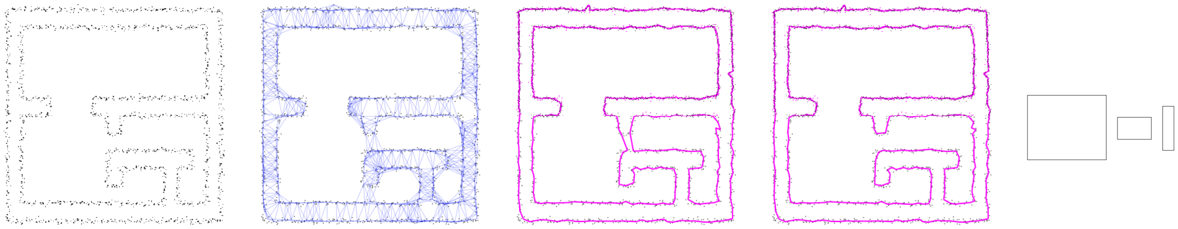
[PMG04] PAULY, M., MITRA, N. J. & GUIBAS, L. J. (2004). Uncertainty and Variability in Point Cloud Surface Data, Point Based Graphics 04.

[SSB05] STEINKE, F., SCHÖLKOPF, B. & BLANZ, V. (2005). Support Vector Machines for 3D Shape Processing, Eurographics 05.

[TAU95] TAUBIN, G. (1995). A signal processing approach to fair surface design. SIGGRAPH 95, pp. 351–358.

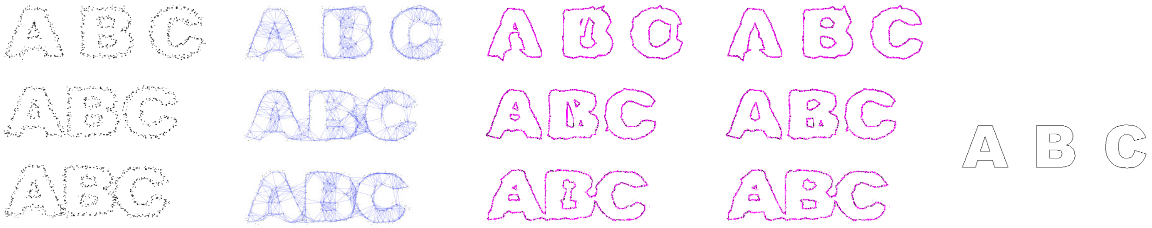
[TB99] TURK, G. & O'BRIEN, J. F. (1999). Shape transformation using variational implicit functions, SIGGRAPH 99, pp. 335–342.

[WJH*07] WAND, M., JENKE, P., HUANG, Q., BOKELOH, M., GUIBAS, L. & SCHILLING, A. (2007): Reconstruction of deforming geometry from time-varying point clouds, Symposium on Geometry processing 07, pp. 49–58.



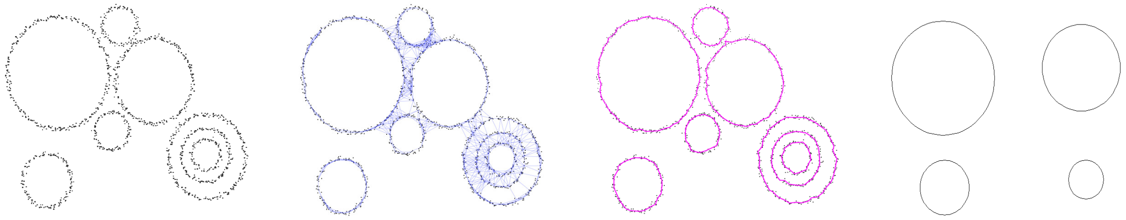
(a) data (b) candidate segments (c) smooth reconstruction (d) empl. learned priors (e) training data

Figure 2: *Floor-Plan 2D example. (a) Data set with 1645 sampling points. (b) Nearest neighbor connection between the keypoints (1151 edges). (c) Reconstructed topology with 285 edges. (d) Reconstruction incorporating training data (e). The reconstruction time is 2.79 sec.*



(a) data (b) candidate segments (c) smooth reconstruction (d) empl. learned priors (e) training data

Figure 3: *ABC example. (a) 2032 sampling points. (b) 1293 Segments defined by nearest neighbor connection. (c) Reconstructed topology with 351 segments. (d) Reconstruction with training data support. The reconstruction time is 3.81 sec.*



(a) data (b) candidate segments (c) reconstruction (d) training data

Figure 4: *Circles 2D Example: Left: Data set with 1346 sampling points. Center: Solution space contains 890 edges. Right: The topologically correct solution with 245 segments. Reconstruction time is 1.97 sec*

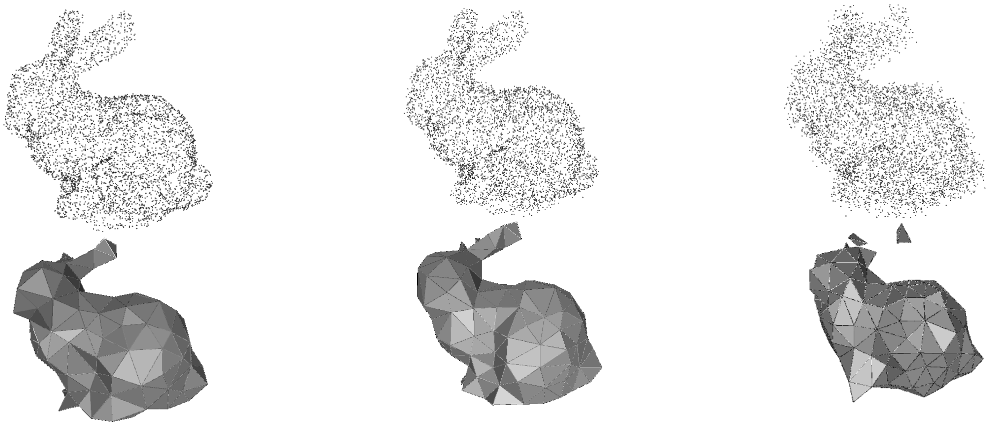


Figure 5: *Stanford Bunny 3D data set. From left to right: different noise levels σ_1=0.01, σ_2=0.02 and σ_3=0.05. Reconstructions times: t_1=146, t_2=831 and t_3=3260 sec.*

	data set	σ *(noise)*	*input points*	*keypoints*	*candidate segments*	*chosen segments*	*comp. time (sec)*
2D	ABC	0.002	2032	353	1315	353	4.29
		0.005	2032	397	1683	397	7.63
	floor plan	0.002	1645	291	1190	291	3.5
		0.005	1645	293	1225	293	4.11
	circles	0.002	1346	243	835	243	1.84
		0.005	1346	245	890	245	1.97
3D	Stanford Bunny	0	5085	153	867	302	98
		0.02	5085	150	779	292	831

Table 1: *Performance of the algorithm for different objects and noise levels. The noise level is given as absolute standard deviation of additive Gaussian noise; all scenes are contained in a unit bounding box.*