Vis Comput (2011) 27: 211–226
DOI 10.1007/s00371-010-0527-x

ORIGINAL ARTICLE

Ellipse-based principal component analysis for self-intersecting curve reconstruction from noisy point sets

O. Ruiz · C. Vanegas · C. Cadavid

Published online: 25 September 2010

Abstract Surface reconstruction from cross cuts usually requires curve reconstruction from planar noisy point samples. The output curves must form a possibly disconnected 1-manifold for the surface reconstruction to proceed. This article describes an implemented algorithm for the reconstruction of planar curves (1-manifolds) out of noisy point samples of a self-intersecting or nearly self-intersecting planar curve C. $C : [a, b] \subset R \to R^2$ is self-intersecting if $C(u) = C(v)$, $u \neq v$, $u, v \in (a, b)$ ($C(u)$ is the self-intersection point). We consider only *transversal* self-intersections, i.e. those for which the tangents of the intersecting branches at the intersection point do not coincide ($C'(u) \neq C'(v)$). In the presence of noise, curves which self-intersect cannot be distinguished from curves which nearly self-intersect. Existing algorithms for curve reconstruction out of either noisy point samples or pixel data, do not produce a (possibly disconnected) Piecewise Linear 1-manifold approaching the whole point sample. The algorithm implemented in this work uses Principal Component Analysis (PCA) with elliptic support regions near the self-intersections. The algorithm was successful in recovering contours out of noisy slice samples of a surface, for the *Hand*, *Pelvis* and *Skull* data sets. As a test for the correctness of the obtained curves in the slice levels, they were input into an algorithm of surface reconstruction, leading to a reconstructed surface which reproduces the topological and geometrical properties of the original object. The algorithm robustly reacts not only to statistical non-correlation at the self-intersections (non-manifold neighborhoods) but also to occasional high noise at the non-self-intersecting (1-manifold) neighborhoods.

Keywords Self-intersecting curve reconstruction · Elliptic support region · Principal component analysis · Noisy samples

Glossary

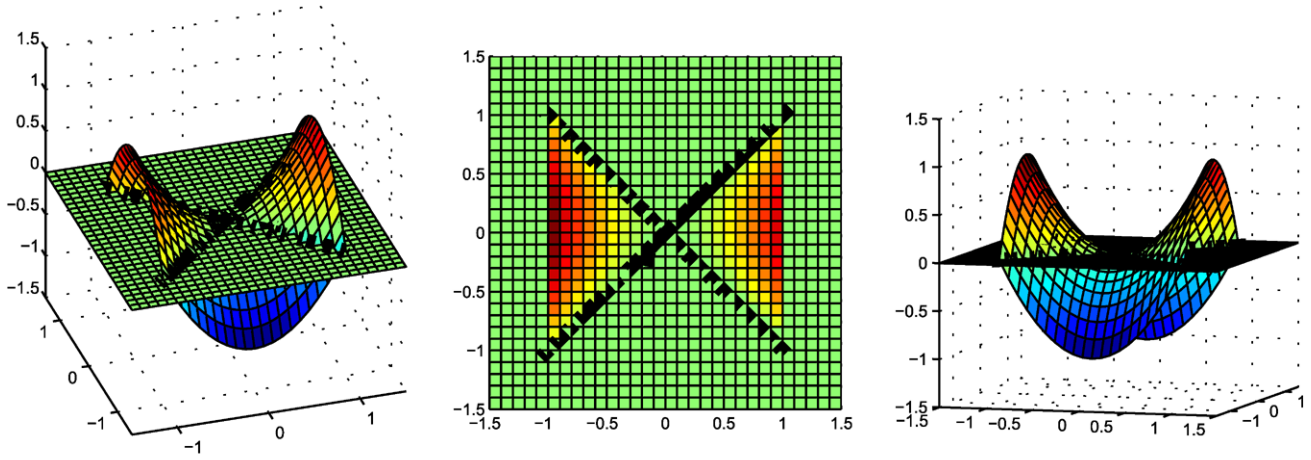
PL: Piecewise Linear.
C: Planar open or closed, possibly self-intersecting or nearly self-intersecting, curve.
$S = \{p_0, p_1, \ldots, p_n\}$: An unorganized noisy point sample of C.
ϵ: Stochastic component of the point sample.
$B(p, r)$: The disk of radius r centered at point p.
$L(\lambda) = p + \lambda * \hat{v}$: Parametric form of the straight line passing through p, directed by the unit vector $\hat{v}$ with signed distance parameter λ.
f_1, f_2: Foci of an ellipse in R^2.
$E(f_1, f_2, \alpha)$: Ellipse $\{p \in R^2 : d(p, f_1) + d(p, f_2) = 2\alpha\}$.
$\rho_{X,Y}$: Linear regression correlation coefficient between variables Y and X.
$[\rho, p, \hat{v}] = pca(S_E)$: Principal Component Analysis of the point set S_E, rendering as a result the linear trend $L(\lambda) = p + \lambda * \hat{v}$ with correlation coefficient ρ.
Q: Queue whose elements are pairs $[p, v]$ formed by a vector v anchored at point p.
$PL\_Curve\_Set = \{c_1, c_2, \ldots, c_m\}$: Set of PL pairwise disjoint curves $c_1, c_2, \ldots, c_m$.

O. Ruiz · C. Cadavid
Department of Computer Science, Purdue University, West Lafayette, IN 47907-2066, USA

O. Ruiz
e-mail: oruiz@eafit.edu.co

C. Cadavid
e-mail: ccadavid@eafit.edu.co

C. Vanegas (✉)
Laboratory of CAD CAM CAE, EAFIT University, Cra 49 7-sur-50, Medellin, Colombia
e-mail: cvanegas@cs.purdue.edu



(a) View 1. (b) View 2. (c) View 3.

Fig. 1 Cut of the hyperbolic paraboloid $z = x^2 - y^2$ with the plane $z = 0$ forms contours which self-intersect at the saddle point $(0, 0, 0)$

1 Introduction

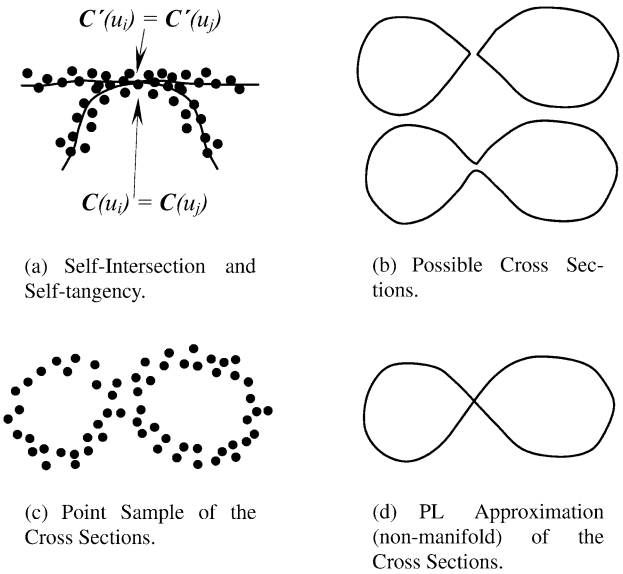
This paper discusses the implementation and results of an algorithm to reconstruct Piecewise Linear (PL) approximations for a possibly self-intersecting or nearly self-intersecting planar curve C sampled with a noisy point set.

By C we mean a function $C : [a, b] \subset R \to R^2$ that is continuously differentiable and regular (i.e. $C'(u) \neq 0$ for all $u \in [a, b]$). C will be said to be *self*-intersecting if there is a finite set $\{u_1, \ldots, u_n\} \subset (a, b)$ such that for each i there is a $j \neq i$ such that $C(u_i) = C(u_j)$ (the $C(u_i)$'s are the self-intersection points). We consider only *transversal* self-intersections, i.e. those for which the tangents of the intersecting branches at the intersection point do not coincide ($C'(u_i) \neq C'(u_j)$).

The cross cuts of a surface might be self-intersecting contours as shown in Fig. 1. Figure 2(a) shows a non-transversal self-intersection with a sample. An ϵ-near self-intersecting curve is one for which there exists a point sample with noise ϵ being identical to the ϵ sample of some self-intersecting curve. In the rest of the article we will simply refer to these as *nearly self-intersecting curves* (omitting the ϵ). The cross section of an object might have a configuration as in the upper or lower parts of Fig. 2(b). A typical noise sample of such cross sections is the set of points $S = \{p_0, p_1, \ldots, p_n\}$ with $p_i \in R^2$ as in Fig. 2(c). Notice that the curves might have any of the forms in Fig. 2(b) or be actually self-intersecting as in Fig. 2(d), and the point sample still looks as in Fig. 2(c).

Notice that for curves as in Fig. 2(d) there is no Nyquist-compliant sample, since the local characteristic dimension δ is zero.

The input to the algorithm implemented in this work is presented in Fig. 2(c). Either of the results in Fig. 2(b) is acceptable for our algorithm, as a legal 1-manifold. Such



(a) Self-Intersection and Self-tangency. (b) Possible Cross Sections. (c) Point Sample of the Cross Sections. (d) PL Approximation (non-manifold) of the Cross Sections.

Fig. 2 Ambiguous noise sample of nearly self-intersecting curves

contours may then be processed for surface reconstruction as in [21] to get an approximation of a closed surface. This process is also conducted in the present work. Figure 2(d) shows a PL approximation of C, although it is obviously non-acceptable because it is a non-manifold. However, reaching it is an important achievement for any curve reconstruction algorithm, and can be easily corrected to obtain either situation in Fig. 2(b).

The sample of the curve C in Fig. 2(c) must respect the Nyquist or Shannon [16, 17, 23, 24] criterion for digital sampling to be able to retain the topology of C. This means that the effective sampling interval $\delta = \delta_n + \epsilon$ (nominal plus stochastic components) must be smaller than half of the minimal detail that the sampling is supposed to preserve.

1.1 1-Manifolds in R^2

M is a 1-manifold in R^2 if for each point $p \in M$ there exists a $\delta > 0$ such that $M \cap B(p, \delta)$ is homeomorphic to the real interval $(0, 1)$. M is said to be a 1-manifold with border if for each point $p \in M$ there is a $\delta > 0$ such that $M \cap B(p, \delta)$ is homeomorphic to either of the real intervals $(0, 1)$ or $[0, 1)$. A set of mutually disjoint closed non-self-intersecting curves is a 1-manifold. A set of mutually disjoint non-self-intersecting curves with at least one of them open is a 1-manifold with border. Informally, a small neighborhood of a point at which a curve ceases to be a 1-manifold looks like three or more semi-arcs emanating from the point. For example, in Fig. 2(d), a small neighborhood of the non-manifold point looks like four semi-arcs emanating from the point.

1.2 PL reconstruction of self-intersecting curves sampled with noise

An nth order PL approximation of a curve C out of a noisy sample of it is a polygonal curve $P = [q_0, q_1, \ldots, q_n]$ which resembles the original curve C up to its nth derivative. A by-product of the process producing P is a parameterization of the contour so recovered, which is fundamental in downstream applications, such as surface reconstruction from cross cuts.

In P the concept of a sequence is central. Many algorithms for curve reconstruction fail to establish such a sequence when they approach the self-intersections of C, exactly because the concept of order is destroyed at such neighborhoods. Our algorithm is able to find the sequence of points forming P, even at the self-intersections. A post-processing is then used to break down P into manifold components.

In the present article the authors attack the problem of self-intersecting or nearly self-intersecting curves (which in the presence of noise are indistinguishable) by using a mutating elliptic support region for the PCA calculation. Informally speaking, near the self-intersections the support region for PCA becomes an ellipse, and far away it is circular. This variation makes the algorithm more robust when facing low correlation coefficients at the intersections.

This article is organized as follows. Section 2 presents a taxonomy of the existing approaches addressing the problem, including previous algorithms developed by the authors. Section 3 proposes improvements to existing algorithms to take into consideration self-intersecting curves, along with the necessary mathematical facts supporting such algorithms. Section 4 addresses the application of the methodology to non-trivial topological cases and presents the results of surface reconstruction from planar slice samples. Section 5 concludes the article and discusses possible future work directions.

2 Literature review

The reconstruction of a curve C out of a noiseless point sample is addressed by relatively abundant literature, relying mostly on graph synthesis techniques. However, since we are interested in Design and Manufacturing applications we must address noisy point samples.

The strategies for the reconstruction of C mainly found in the reviewed literature are: (1) Medial axis calculation; (2) Scalar field calculation, with: (2.1) Radial basis functions, (2.2) Differential equations (e.g. level sets); (3) Statistical estimation by Principal Component Analysis, including: (3.1) Noising/de-noising of the point set, (3.2) Straight segment synthesis, (3.3) Parametric curve synthesis (Bézier, Spline, NURBS); (4) Probabilistic estimation of topological properties; (5) Probabilistic estimation of geometrical properties.

Techniques transversal to many of the approaches mentioned above are: (a) minimization techniques, (b) graph theory, (c) probabilistic and statistical estimation, (d) Delaunay–Voronoi based methods.

In the reviewed literature, the vast majority of the articles do not address the issue of (nearly) self-intersecting curve reconstruction out of point samples. The two references reporting such advance do not explain how their methods effectively deal with such a feature.

In the consulted literature there is a general absence of formal analysis for the computational complexity of the proposed algorithms. When present, such a discussion only addresses average cases and central *time* expenses, ignoring the *space* complexity of the collateral data structures and the *time* spent in building them.

2.1 Medial axis

[10] and [8] explore the recovery of a Principal Graph underlying a 2D point sample (e.g. a character meant to by pen strokes). The authors set up a numerical optimization algorithm that balances two competing criteria: (i) the inclusion in the graph of as many pixels as possible of the ones present in the stroke, and (ii) the minimization of the medial axis curvature. Since this algorithm aims at character recognition, its final result is not required to be a 1-manifold. Therefore, self-intersections are permitted (like in the "H" or "8" characters). In our case, the final result of the reconstruction must be a set of disjoint non-self-intersecting curves, and therefore one must meet higher requirements than the ones met by [8] and [10]. The algorithm implemented in [10] and [8] finds an approximation to the medial axis of the black pixel region. The complexity of the algorithm is estimated by the authors in $O(N)$, where N is the number of black pixels in the image. This estimation must be carefully interpreted since it only takes into account a part of the

process. For example, only finding a medial axis approximation has a minimal complexity of $O(N^2)$. In addition, the proposed strategy requires collateral data structures whose time and memory expenses significantly increase the cost of the whole process of curve reconstruction.

[25] discusses the synthesis of the skeleton of a 2-manifold sampled with *oriented points*. An *oriented point* contains the (x, y, z) coordinates and the vector normal to the surface at such point. The manifold is constructed with cylindrical branches meeting at joint neighborhoods. The point sample might be incomplete, according to the authors. The point sample is partitioned in quasi-planar sections. Each point subset of the partition must look like the section of a cylinder, according to the basic assumptions on the manifold being sampled. Each cross section of the cylinder has a *statistical* center. The sequence of such centers contributes to the skeleton. At the joints, there are no such skeletons. At such regions several processes are applied to obtain a line-like structure: smoothing, thinning, re-centering, joint collapsing and redistribution of point samples. This process requires intensive user control. Notice that the skeleton is not a manifold in general. Therefore, the proposed algorithm is not required to neutralize the non-manifold neighborhoods. The authors do not discuss the complexity of their algorithm.

2.2 Scalar fields

2.2.1 Radial basis functions

[20] presents a method to establish a likelihood map or scalar function in R^2 around a noisy point sample. The scalar function records a high value for (x, y) if it is close to a sampled 2D curve C. After a likelihood map is calculated (similar to Fig. 3(b)), a PL approximation is initialized in the form of a topological circle on R^2, and then allowed to drift to settle on the highest values of the likelihood map. The paper does not discuss self-intersecting curve point samples, disconnected curves, initial size or position of the circle, or computational complexity of the method.

[9] presents a definition of an implicit surface over a noisy point cloud using weighted least squares based on a geometric proximity graph. Self-intersections are not addressed. In this reference, computing a Close Pair Shortest Paths (CPSP) table among N points in R^3 is reported to consume $O(N)$ computing time and $O(N)$ storage space. Such predictions do not count the time in building the collateral (breath-first, depth-first) data structures. No accounting is devoted to the administration of collateral data structures or pre-processing time.

2.2.2 Fitting of $y = f(x)$

[1] fits polynomials $h(x)$ to a series of points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$ such that $h(x_i) \in [y_i - \delta, y_i + \delta]$. Important limitations of this reference are: (1) the majority of applications in applied computational geometry deal with curves in R^2 which are not the graphs of *functions*. For example, a closed contour in R^2 cannot be expressed as $y = h(x)$. (2) No applications are given in the article. (3) The authors do not discuss the complexity and scope of their solution.

2.2.3 Reconstruction by differential equations and level sets

Consider a noisy point set S in R^2 sampled on a regular curve C. There is abundant literature that seeks to recover an approximation to C as a solution of a differential equation stated on a domain Ω which contains S. The goal of such methods is to synthesize an implicit function $f : R^2 \to R$, solution of a differential equation, and the sought curve C is the implicit curve $f(x, y) = 0$.

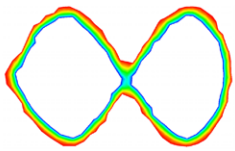
From the reviewed literature [13, 18, 19, 30, 31] we may conclude the following: (1) The definition of the Ω region covering the set S is already an open problem. However, a convenient informal definition of Ω would be the tape-shaped polygon covering S (see Fig. 3). (2) To solve the differential equation it is essential to draw geometric information from the point sample itself. For example, level set methods require a vector field $v : R^2 \to R^2$ normal or tangent to C at every point to be able to reconstruct C. As an effect, self-intersecting curves cannot be recovered in this manner since in such curves the tangent/normal fields are undefined at the self-intersections. (3) Whichever solution $f : R^2 \to R$ can be found by solving differential equations, it must be kept in mind that any curve C recovered as an iso-level set of f *will be a closed one*. This implies that open curves C cannot be recovered by differential equation methods, unless additional manipulations on the domain Ω (not reported yet) are introduced. (4) Once a function f has been estimated by solving the differential equation, the value k representing the iso-curve $f(x, y) = k$ that approaches C must be guessed (Fig. 3(c)), and such an iso-curve might not be a 1-manifold. (5) $f(x, y) = k$ might contain disconnected curves even if C was originally connected. (6) The computationally obtained solution to a differential equation f is a function whose domain is a grid of points. Passing from f to iso-curves $f(x, y) = k$ clearly requires an additional process [2–4]. For all these reasons, finding f as a solution to a differential equation still needs a significant effort before it can be considered as a reliable tool for building 1-manifolds out of the point set S.

2.3 Statistically based methods

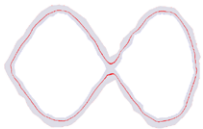
2.3.1 Noising/de-noising of point set

[5] attack the problem of fitting PL curves to noisy point samples by computing a sequence of new point sets having

(a) Point Sample of the Self-intersecting Curve.



(b) Iso-curves of the solution to the Differential Equation.



(c) Loci of $f(p) \approx 0$. Dark color represents a 2D region, not a curve.

Fig. 3 Curve reconstruction with differential equations [22, 27]

less noise than the initial point set and less points. This is done by searching thin rectangles normal to the local curve tangent for sample points and by driving the sample to the expected value of the curve. When the point set is sufficiently thin, the actual PL approximation to C is computed using a crust algorithm (in this case the NN Crust by [6]). The algorithm is guaranteed to converge if a sufficiently good point sample of C is available. We must point out that such a sample does not exist for self-intersecting curves, as the Nyquist criterion cannot be met. This algorithm is misled when the curve gets close to itself because the thinning of the point set about the likely curve locus has two or more attractors. [5] discusses the probabilities of the obtained curve being homeomorphic to the original one. The authors neither discuss computational complexity nor present application examples.

[14] presents an algorithm that takes a noise-free sample of a non-self-intersecting curve in R^2. The algorithm adds noise in the point sample, in the direction perpendicular to the originally sampled curve. The algorithm eliminates the noise by replacing the points falling in a circle by the center-point of a segment joining the most extreme points inside the circle. After the noise is removed, the point set is fed to a Relative Neighborhood Graph, derived from the Delaunay triangulation. Shortcomings of this approach are: (1) Noise is added to a point set that is originally noise-free. (2) The original point set is filtered, with high frequencies removed. (3) The noise removal pre-processing costs $O(N^3)$ computation time. (4) The article presents results for the enlargement of the point set but it does not do so for the curve reconstruction itself. (5) The article does not discuss complexity at any point.

2.3.2 *Synthesis of straight segments*

[11] presents a least-squares algorithm to approximate a set of unorganized points with a simple 3D curve without self-intersections. This algorithm uses an Euclidean Minimum Spanning Tree (EMST). The algorithm performs thinning on the point cloud before calculating its PL approximation. No discussion of computational complexity is presented.

[28] presents a PL approximation of a planar curve C whose sample S has noise. A set of straight segments is accommodated on the region defined by the sample, with each segment being locally tangent to C. A second part of the algorithm defines an order on the straight segments and threads the tail of one with the head of the next one. This approach cannot define how a self-intersecting curve is handled, since no straight segment and no *tail*, *head* or *next* segment can be defined at the self-intersection, due to the lack of correlation in the local point set. No discussion of computational complexity is presented.

2.3.3 *Synthesis of parametric curves*

[29] fits B-Splines to a set of noisy point sets using curvature-based squared distance minimization. The control points q_i of a Spline curve $P(t) = \sum_{i=1}^{M} B_i(t) * q_i$ are set such that the summation of square distances between $P(t)$ and the point sample is minimized (Square Distance Minimization, SDM). Limitations of this algorithm are: (1) It specifically excludes self-intersecting curves. (2) A (time or space) complexity discussion is absent. (3) Both the sampled curve and the recovered curve are required to be twice differentiable. (4) Calculation of the distance from a point to a parametric curve is expensive since the latter normally has parametric degree larger or equal than three. This makes the SDM an expensive method.

[12] presents an algorithm which seeks to fit B-Spline curves to a set of noisy points in R^2 on which a Euclidean Minimal Spanning Tree (EMST) is calculated. The algorithm uses a band-shaped support region to collect point subsets whose Principal Component Analysis trend determines the local tangents to the sought B-Spline curve. The authors claim that the algorithm handles sharp features and self-intersections of the curves. However, no clear algorithm is given to handle such situations. A fundamental flaw of such an algorithm is that a point sample of a 2D curves does not in general have the topology of a *tree* since cycles exist naturally in such sets. In addition, the algorithm uses a significant amount of heuristic constants that are not discussed in the paper and whose set values are not specified. The B-Splines are naturally smooth in spite of the fact that the point set is sampled from a non-smooth 2D curve. The paper does not discuss the computational complexity of the algorithm.

2.4 Probability of topological properties of the curve

[15] proposes (without actual implementation or tests on data) an algorithm to probabilistically estimate topological properties of a manifold out of a noisy sample of it. [15]

specifically avoids the estimation of topological properties in the case of self-intersections. In this reference there is no comment on the computational expenses of the proposed algorithm.

2.5 Probability of geometrical properties of the curve

In [26] the local neighborhood of 3D curves in the space is sampled with noise σ and sampling density ρ. A local coordinate frame is associated with each neighborhood and the point sample is used to diagnose the curvature κ and torsion τ of the 3D curve. The angle between the actual tangent vector and the PCA-estimated tangent vector is found and plotted as a function of the radius r of the PCA support region. The value of the deviation angle varies with respect to the local curvature κ, noise σ, sampling density ρ and support region radius r. The article does not give a self-tuning algorithm (the radius is given as an absolute number), making the results unusable when the scale of the point set changes. The article does not discuss the computational complexity of the algorithm.

2.6 Conclusion of literature review

The reviewed literature presents some salient features: (1) Implicit function calculations require large computational expenses. (2) Medial axes' methods produce inherently non-manifold constructs. (3) Curve synthesis as solution of Differential Equations is not adequate for the reasons given in Sect. 2.2.3. (4) The fitting of higher degree parametric curves (Bézier, Spline, NURBS) requires non-linear minimizations at every stage of the construction, which implies large computation costs.

We conclude that an explicit form of C, in Piecewise Linear form, is cheaper to determine and is acceptable for subsequent applications (in contrast with implicit forms). To find an explicit PL form of C we implement a method which is sensitive to the proximity of the self-intersection. In such a locality the point set to be fed to a PCA algorithm is the one included in ellipses rather than in disks. As it will be shown, such a variant allows the overcoming of the intersection neighborhood in a more robust manner.

Our ellipse-PCA algorithm will be tested on slice samples of C^2 2-manifolds. As shown in Fig. 1, this is one of the many possible scenarios where self-intersecting or nearly self-intersecting curves are present. As such cases are compounded by noise (present in all industrial sensors), we consider that efforts in this area are useful for the computer-aided design and manufacturing communities.

3 Methodology

The algorithm implemented in this article is based on statistical approximation of the local tangent of a curve C sampled with a noisy point set. The algorithms based on Principal Component Analysis do not perform well in curve self-intersection regions because the linear trend is lost there. To avoid this effect, we used a directional (elliptic) support region for the PCA algorithm. The ellipse becomes sharper as the linear trend of the point sample degenerates (for example, at self-intersections). The elliptical support region has major axis in the direction of the last reliable vector tangent to the curve. This region excludes point samples in the direction perpendicular to such a tangent, thus ignoring the confusing trends at the self-intersection. In this manner the algorithm overcomes self-intersection regions and continues with the PL approximation of C.

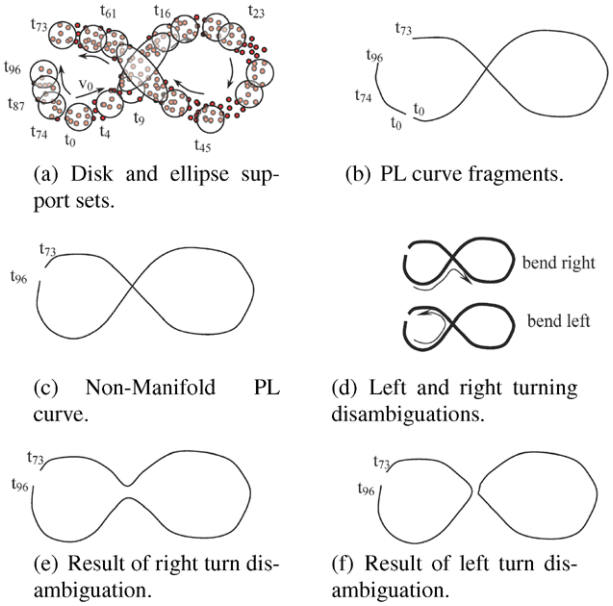


Fig. 4 Execution of ellipse-based PCA algorithm

Figure 4(a)–(c), displays an intuitive functioning of the algorithm implemented, applied to the point sample of an open, self-intersecting curve C (Fig. 4(a)). To simplify the drawing, the point sample is suppressed from some figures, showing only the pursued PL approximation for C. Figure 4(a) shows that the algorithm starts in the neighborhood t_0, with the direction v_0, with a circular PCA-support region. The local approximation travels following the neighborhoods $t_4, \ldots, t_9, \ldots, t_{73}$ (subscripts only indicate a *supposed* number of iterations). At the self-intersection, the PCA-support region becomes elliptical, so the algorithm is capable of crossing this zone, whose correlation coefficient ρ is fundamentally low if a circular support region had been used. The algorithm proceeds until it reaches t_{73}, where it finds a dead end, meaning that a connected subset of C has been found. The algorithm then revisits t_0 with direction $-v_0$, rendering the sequence $t_0, \ldots, t_{74}, \ldots, t_{87}, \ldots, t_{96}$ (Fig. 4(b)).

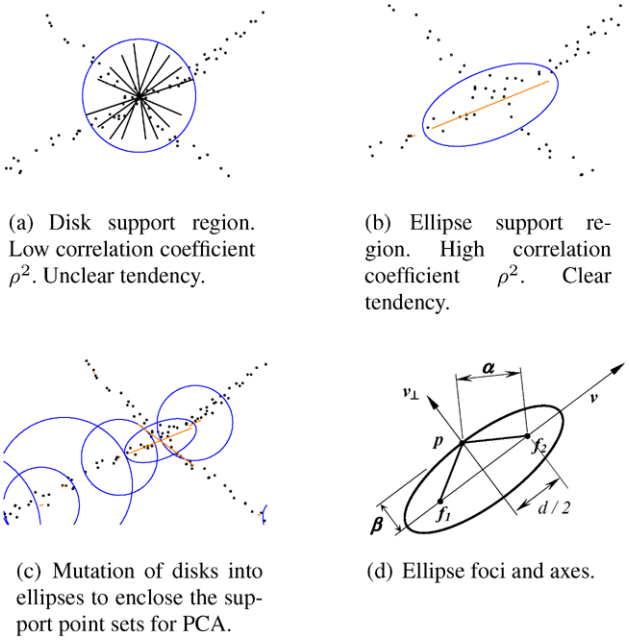
The post-processing part of the algorithm is fully known in computational geometry: it merges the PL approximations $[t_0, \ldots, t_{73}]$ and $[t_0, \ldots, t_{96}]$ into $L = [t_{73}, \ldots, t_0, \ldots, t_{96}]$, and then splits the self-intersecting PL curves determined by L as per the decisions in Fig. 4(d), resulting in the 1-manifolds in Figs. 4(e) or 4(f). Either, right or left splitting produces topologically correct and geometrically different results. Since the (noisy, self-intersecting) sample does not allow for a canonical choice of either splitting (left or right), such a decision in the post-processing stage is one of mere convention.

3.1 Measure of goodness-of-fit for principal component analysis

If a curve C has local curvature radius r and is point-sampled, there are upper and lower bounds in the length of a linear segment $\overline{ab}$ which statistically approaches C at the point $C(u)$. If $\overline{ab}$ is too long, the segment will not correctly approach the curve C at $C(u)$. If $\overline{ab}$ is too small, only few sample points will be available to fit the segment $\overline{ab}$. In both cases, goodness-of-fit of the linear approximation is degraded. If C is planar, the Principal Component Analysis may be evaluated by using the linear regression correlation coefficient $\rho = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E((X-\mu_X)(Y-\mu_Y))}{\sigma_X \sigma_Y}$ with $\rho \in [-1, 1]$. $|\rho| \approx 1.0$ and $|\rho| \approx 0.0$ are associated with good and poor linear correlation, respectively. Caution must be exercised because linear regression parameters (m, b in $y = mx + b$) are dependent on the particular coordinate system. Figure 6 shows that for the same level of noise the correlation coefficient ρ^2 between x and y varies as m does. If we wish to compare *linearity* of point sets using the ρ^2 values, we should first achieve a fixed m (for example $m = 1$) by rotating the point set and only then calculate its ρ^2 value. In this manner we compare m-constant ρ^2 values of different point sets. On the other hand, an advantage of using the correlation coefficient to grade the linear regression is that the ρ^2 value is bounded ($\rho^2 \leq 1$). Our approach is to use the linear regression with its correlation coefficient by following these steps: (a) Calculate a tentative linear trend $y = m_0.x + b_0$ for the local point set; (b) Rotate the local point set to get a slope of 45° ($m \approx 1$); (c) Calculate ρ with the standard linear regression formulas; (d) Rotate back the measured m if needed. A number of iterations (limited to 10) is used to improve the value ρ by varying the shape of the ellipse enclosing the local point set. In this manner, we use the advantages of 2D linear regressions and neutralize its dependence on m. This heuristic worked correctly, as discussed in the Results section.

3.2 Circular vs. elliptical support regions

Previous algorithms for curve reconstruction avoid addressing the topic of self-intersecting curves due to the effect



(a) Disk support region. Low correlation coefficient ρ^2. Unclear tendency.

(b) Ellipse support region. High correlation coefficient ρ^2. Clear tendency.

(c) Mutation of disks into ellipses to enclose the support point sets for PCA.

(d) Ellipse foci and axes.

Fig. 5 Points and ellipses in a self-intersection region

shown in Fig. 5(a). At the self-intersection neighborhood, the identification of the local tangent becomes difficult and the curve reconstruction goes astray. This happens because the PCA analysis applied to a star-shaped point set will render lines in any direction of the R^2 plane, accompanied of the correlation coefficient ρ being very low.

The algorithm reported in this article detects such low correlation regions and varies the shape of the support region for the PCA from round to elliptic (see Fig. 5(b)), thus incrementing the correlation coefficient ρ. In our algorithm, ellipticity increases together with least square fitting error. The elliptic region has the major axis in the direction of the last reliable curve tangent v identified in the previous iteration (Fig. 5(d)). This support region has the advantage of ignoring distracting points by having a tight span in the $v_\perp$ direction, orthogonal to v. The ellipse is defined as the locus of points p such that $|p - f_1| + |p - f_2| = 2\alpha$. Our algorithm sets $\alpha = 5 * \delta$, where δ is the effective sampling interval of the device. This value is set to approximately include 10 sampled points in each local Principal Component Analysis. Such a heuristic has worked in a stable manner in our implementations. Therefore α is a known value that is fixed in the algorithm.

3.3 The heuristic to overcome intersection places

In general, an ellipse in R^2 can be specified by the position of its foci f_1 and f_2 (Fig. 5) and the length of its major semi-axis α as $E(f_1, f_2, \alpha) = \{p \in R^2 : |p - f_1| + |p - f_2| = 2\alpha\}$. In any ellipse, if d is the distance between the foci, then $\beta^2 = \alpha^2 - (\frac{d}{2})^2$.

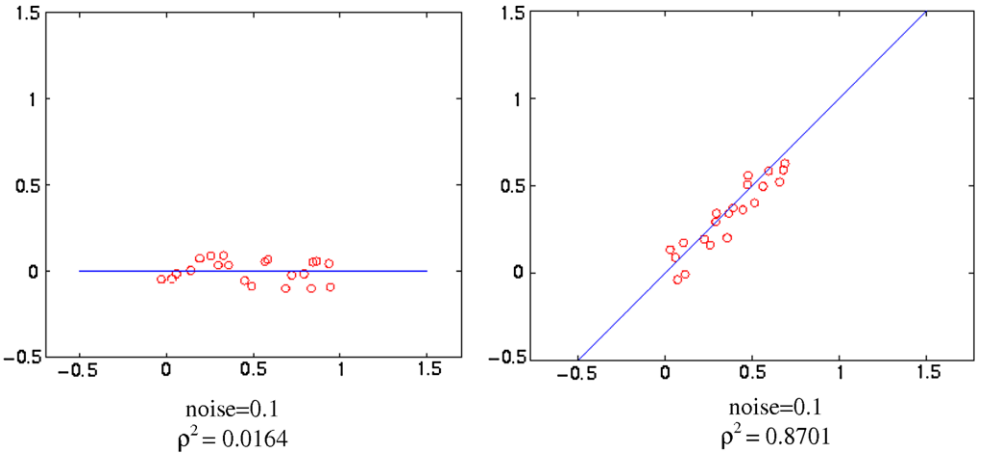
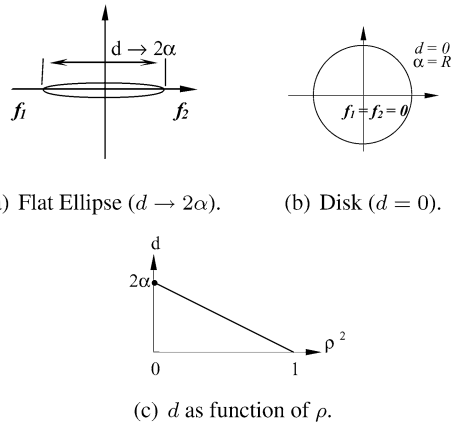


Fig. 6 Dependency of correlation coefficients with slope m



(a) Flat Ellipse ($d \to 2\alpha$). (b) Disk ($d = 0$).

(c) d as function of ρ.

Fig. 7 Heuristic rule which increases d (i.e., flattening the ellipse, $d \to 2\alpha$) as the correlation coefficient deteriorates ($\rho \to 0$).

Our claim is that in a well-defined curve point sample a PCA circular support region would work fine. If the point set deteriorates, the support region must become an ellipse. If the linear correlation of the point set is poor ($\rho^2 = 0$), we would like to have a strongly elliptical support region (large d, Fig. 7(a)). If the linear correlation is good ($\rho^2 = 1$), a circular disk ($d = 0$, Fig. 7(b)) would provide a convenient support region. We propose a decreasing function $d = 2\alpha(1 - \rho^2)$, as in Fig. 7(c). Therefore, d ranges between 0 (good PCA correlation) and 2α (poor PCA correlation).

3.4 Reconstruction algorithm

The implemented algorithm takes a point set as in Fig. 4(a) and returns a set of PL curve fragments as in Fig. 4(b). This result is the fundamental one, because joining the PL fragments of C in PL_Curve_Set in a manifold manner is a standard procedure.

The algorithm is based on the heuristic proposed in Sect. 3.3, which gradually mutates a circular into an elliptical support region for the Principal Component Analysis. A simplified version of it is presented as Algorithm 1.

The algorithm contains three nested WHILE iterations. The invariant of the WHILE in line 8 is that a set of PL curve fragments (PL_Curve_Set) has been synthesized and that there exist neighborhoods of unused points which have not been considered yet ($Q \neq \emptyset$). The invariant of the WHILE in line 12 is that a local PL fragment ($local\_C$) is being threaded as long as a well-defined tangent is identified along it. The invariant of the innermost WHILE (line 18) indicates that in a particular neighborhood, a PCA based on circular support regions has produced a low ρ indicator. Therefore, the support region is gradually flattened until either ρ surpasses a threshold or a specified number of trials is reached. In the first case, the algorithm proceeds to the next (in the direction of v) neighborhood. In the second case, the algorithm recognizes the fact that no clear tangent has been identified and declares the local curve as finished. This WHILE iteration implements the heuristic discussed in Sect. 3.3.

The elements of the queue Q (line 5) have the form $[p, v]$ where p is a point near C and v is a (unit) vector tangent to C near p. Notice that $-v$ is also tangent to C at p. Each element of Q indicates a place and direction for traversing the sample S for the recovery of a portion of C. If the queue Q is empty, the algorithm finishes. The built-in function $[\rho, p, v] = pca(S)$ is used in order to perform the Principal Component Analysis of the point set S giving as a result the trend v, the center of mass p, and the correlation coefficient ρ.

The algorithm builds each curve fragment or local curve $local\_C$ as long as there is a clearly computable vector tangent to C at each point p. If the tangent is clear, a circular-supported PCA calculation is sufficient (lines 15, 16) to determine it. Otherwise, an ellipse-supported PCA is attempted (lines 18–24). If the ellipse-based PCA manages to overcome the self-intersection, the algorithm continues (lines 26, 27) completing the curve fragment $local\_C$. If the elliptic support regions at both extremes of $local\_C$ lead the PCA to fail identifying a clear tangent vector, the algorithm stops processing the current fragment $local\_C$ (line 29). In

Algorithm 1 PCA-based reconstruction algorithm using ellipses.

1: Comment: S is the sample of C with noise.
2: Comment: r is set to the sampling noise plus sampling distance.
3: Let p be any point in S such that $S \cap B(p,r)$ presents a correlation coefficient $\rho \approx 1$.
4: $[\rho, p_t, v_t] = pca(S \cap B(p,r))$
5: $Q = queue([p_t, v_t])$
6: $Q = add(Q, [p_t, -v_t])$
7: $PL\_Curve\_Set = [\,]$
8: **while** $Q \neq \emptyset$ **do**
9: $[p, v] = discharge(Q)$
10: $local\_Cu = [\,]$
11: clear_tangent=TRUE
12: **while** clear_tangent **do**
13: $local\_C = [local\_C\ ,\ p]$
14: $p_t = p + \lambda * v$
15: $local\_S = S \cap B(p_t, r)$
16: $[\rho, p_t, v_t] = pca(local\_S)$
17: $Num\_Trials = 1$
18: **while** $(Num\_Trials < Max\_Trials)$ and $(\rho < Lower\_Bound)$ **do**
19: $d = 2\alpha(1-\rho^2)$
20: $f1, f2 = p_t \pm \frac{d}{2} * \hat{v}$
21: $local\_S = S \cap E(f1, f2, \alpha)$
22: $[\rho, p_t, v_t] = pca(local\_S)$
23: $Num\_Trials = Num\_Trials + 1$
24: **end while**
25: **if** $(Num\_Trials < Max\_Trials)$ **then**
26: $v = v_t$
27: $p = p_t$
28: **else**
29: clear_tangent = FALSE
30: Let p be an unused point in S whose neighbor point set inside a disk $B(p,r)$ has $\rho \approx 1$.
31: **if** p is found **then**
32: $[\rho, p_t, v_t] = pca(S \cap B(p,r))$
33: $Q = add(Q, [p_t, v_t])$
34: $Q = add(Q, [p_t, -v_t])$
35: **end if**
36: **end if**
37: **end while**
38: $PL\_Curve\_Set = [PL\_Curve\_Set, local\_C]$
39: **end while**
40: Comment: PL_Curve_Set is the set of PL fragments approximating C.

this situation, the algorithm seeks unused neighborhoods of the point set that may originate another fragment *local_C* when taken as seed in later iterations (line 30). If such neighborhoods are found, they are input into the queue Q (lines 32–34).

3.5 Complexity of the algorithm

Let us assume that the number of points in the sample is N. In Algorithm 1, either one of lines 21 or 22 contains instructions whose worst-case cost is $O(N)$. Since such instructions are inside threefold nested WHILE loops whose worst-case complexity is $O(N)$ each, we conclude that the *worst-case* complexity for such an algorithm is $O(N^4)$. It is important to observe that in our approach no additional memory or time resources are spent in building or maintaining collateral data structures or in pre-processing the data.

In this regard, the literature reviewed is uniformly incomplete in that run-time complexities are given without reporting resources devoted to (a) collateral data space and (b) preprocessing. Since our evaluation $O(N^4)$ is a worst-case estimate and specifically rules out the need of expenses (a) and (b) above, it is not comparable with other evaluations which concentrate on *expected* cases and neglect to take into account the expenses caused by (a) and (b) (see the Sects. 2 and 2.6).

Integration of PL fragments

This part of the algorithm is well known in computational geometry and it is not dominant in terms of complexity, as compared with Algorithm 1. The statement of this post-processing is as follows. Given an unordered set of PL curve fragments $PL\_Curve\_Set = \{c_1, c_2, \ldots, c_m\}$ that approximate the point set S (Fig. 4(b)), two steps are required: (1) the joining of c_i and c_j when their endpoints are closer than a distance δ_s (Fig. 4(c)), and (2) the splitting of the paths resulting from (1) to avoid self-intersections, by using the decision criteria in Fig. 4(d). The final result appears in Figs. 4(e) and (f). The processes (i) and (ii) considered together have complexity $O(N^2)$.

4 Results

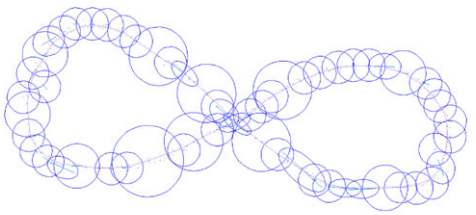
Figure 8 shows the functioning of the proposed algorithm, applied to a closed curve with self-intersections. It can be seen that the circular support regions ($|f_1 - f_2| \to 0$) at manifold neighborhoods become flattened ellipses ($|f_1 - f_2| \to 2\alpha$) at non-manifold neighborhoods.

Outliers (points sampled with unusually large sampling noise) do not participate in the execution. The algorithm is robust in this sense, since it flattens the ellipse as a response to the inclusion of such outliers in the PCA. As a result, such points are expeditiously ignored. The whole algorithm stops when most of the points (near 100%) have been considered

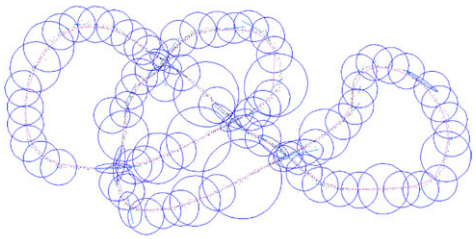
in at least one ellipse or disk. The tests run provide strong evidence that this stopping criterion does not affect the efficacy of the algorithm.

The implemented algorithm was applied to three (*Hand*, *Pelvis* and *Skull*) data sets. Such data sets correspond to cross-sectional parallel point samples applied to an object which might have internal cavities. The point samples are used as input to the curve reconstruction algorithm discussed in the present article. The PL curves resulting from our algorithm are then fed to a surface reconstruction algorithm which accepts parallel planar contour sets ([21] or [7]), as a means to prove their topological and geometrical correctness.

A precision must be made to explain what a *failure* of the implemented algorithm means: we consider that the algorithm fails if it is not able to fit one PL curve in a local neighborhood of the point sample (e.g. Fig. 16(f)). In contrast, if the algorithm implemented fits many disconnected or non-manifold PL curves to the point sample (Figs. 14(j) or 16(d)), such situations are corrected by running a post-processing which joins the individual PL curves (Figs. 4(b) and 4(c)) and then splits them at the self-intersections, forming disconnected closed contours (Figs. 4(e) and 4(f)). The final product is a set of PL curves (1-manifolds), perfectly suitable for downstream applications (for example, surface reconstruction). Therefore, these two outcomes are not to be considered as failures of the algorithm.



(a) One self-intersection.



(b) Four self-intersections.

(c) Detail of self-intersection.

Fig. 8 Ellipse-PCA processing self-intersecting curves

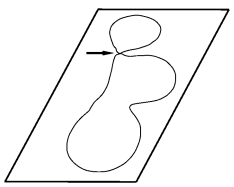
4.1 Data set 1. Hand

In surface reconstruction from slice samples it is not uncommon to have one or more (usually non-consecutive) missing slice samples. In such a case, it is appealing to replace the missing slice sample i by the projection of the point data from slices $i-1$ and $i+1$ onto the plane corresponding to it. An example of such a projected point set is depicted in Fig. 9(a). It must be pointed out that such a point set presents the additional difficulty of having noise stemming from the point projection, besides the basic sampling noise. Figure 9(b) presents the result of the application of Algorithm 1 to such a point set. A standard algorithm for separation of non-manifold curves into manifold ones produces the separated contours (Jordan curves in R^2) in Fig. 9(c).

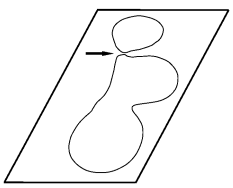
Figures 10(a) and 10(b) present a zoom on particular details of Figs. 9(b) and 9(c), respectively. Figure 10(a) presents a neighborhood of self-intersecting PL curves obtained with Algorithm 1. Such neighborhood with the self-intersection removed is shown in Fig. 10(b). Additional results of self-intersecting cross cuts of the *Hand* data set are displayed in Fig. 11.

Algorithm 1 was tested on the *Hand* data set, made of slice noisy point samples of an object. The result of applying Algorithm 1 to all slices of such a data set is displayed

(a) Example of doubly-noised input point set.

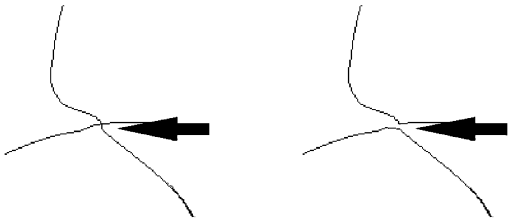


(b) Result of ellipse-based PCA. PL Curves are not 1-manifolds.



(c) 1-manifoldness condition after post-processing of self-intersecting PL curves.

Fig. 9 *Hand* data set. Noisy point set **(a)** along with its processing **(b)** and post-processing into a disconnected 1-manifold **(c)**



(a) Detail 2. Self-intersecting PL Curve. (b) Detail 2. Broken Self-Intersection.

Fig. 10 Detail of broken self-intersection of Fig. 9

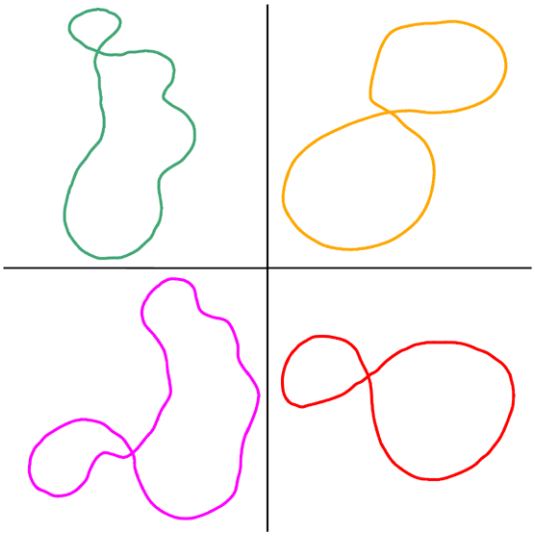
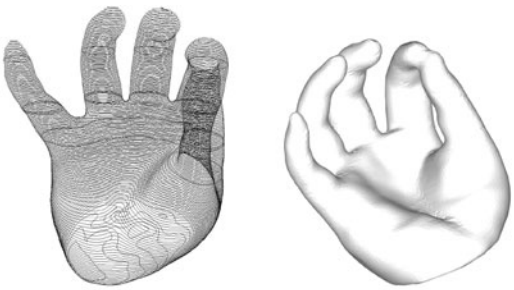


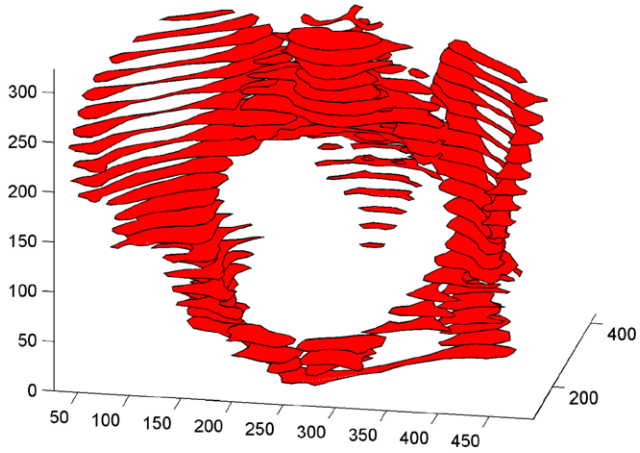
Fig. 11 Additional examples of self-intersecting contours in the *Hand* data set



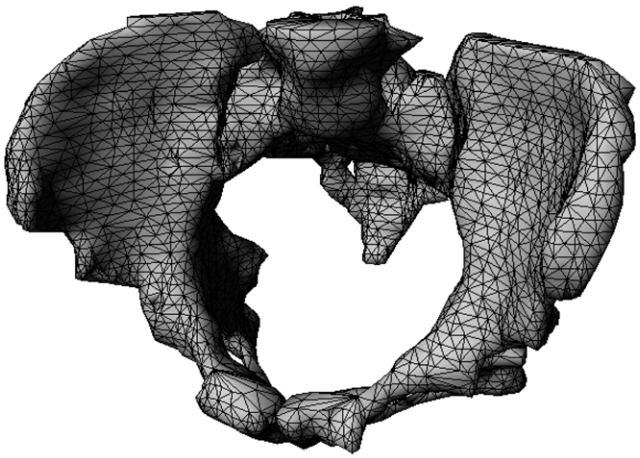
(a) PL contour approximation results. (b) Complete model.

Fig. 12 Algorithm results for the *Hand* data set. Rendered surfaces

in Fig. 12(a). The slices containing self-intersections are the darker ones. The PL contours belonging to the slices were then fed to well-known algorithms ([21] or [7]) to reconstruct the surface. Fig. 12(b) presents the surface for the *Hand* point set including the whole set of cross sections.



(a) Recovered cross sections.



(b) Reconstructed surface.

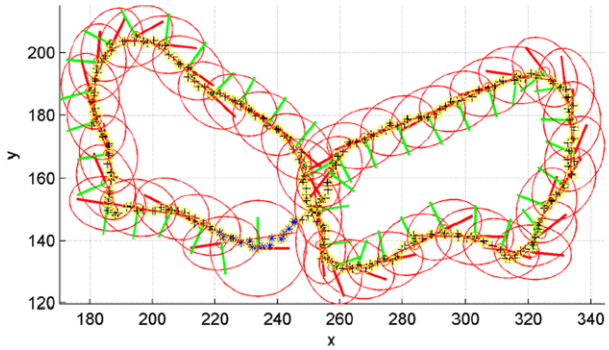
Fig. 13 Reconstructed contours and surfaces for the *Pelvis* data set

4.2 Data set 2. Pelvis

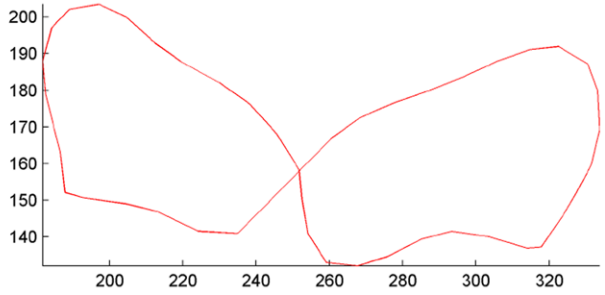
To further illustrate here the robustness of the proposed method, a nearly self-intersecting contour set was extracted from the *Pelvis* data set (Fig. 13) and was added with noise levels $[1\delta_n, 2\delta_n, 3\delta_n, 4\delta_n, 5\delta_n, 6\delta_n]$ (δ_n is the nominal sampling interval). The algorithm was then run using such point sets (see Fig. 14). The ellipse sequences of our algorithm are displayed in the left column, while the recovered contours (before splitting) appear in the right column.

Notice that the algorithm is able to fit *one* PL curve to the whole point set at once for noise levels $[1\delta_n, \ldots, 4\delta_n]$, showing stable performance for such cases. For noise levels $5\delta_n$ or $6\delta_n$ the performance of the algorithm degrades. It fits several PL curves to the point set, which must be then integrated as in Figs. 14(b) and 14(c). Such actions are discussed in the section "Integration of PL Fragments."

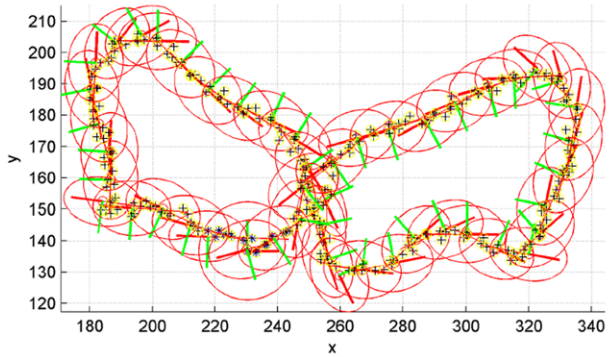
Notice that the strategy proposed in the present article further processes the intermediate results for cases $5\delta_n$ or $6\delta_n$, as these results are not considered to be topologically



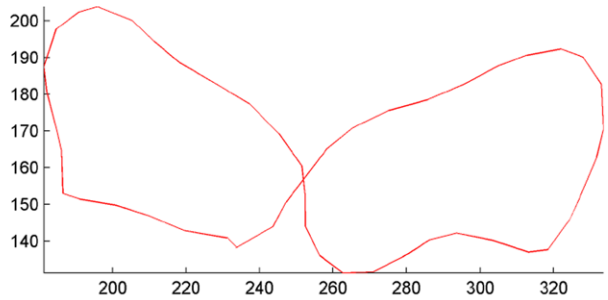
(a) Ellipse sequence. Noise=$1\delta_n$.



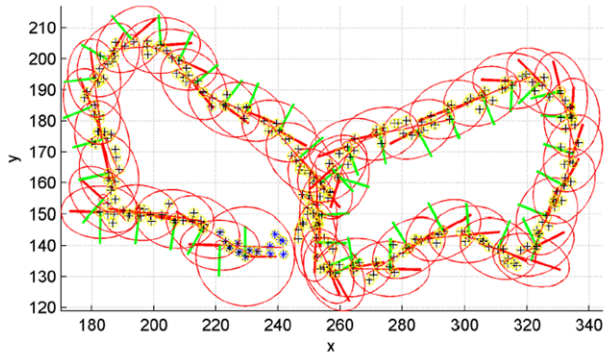
(b) Contours before splitting. Noise=$1\delta_n$.



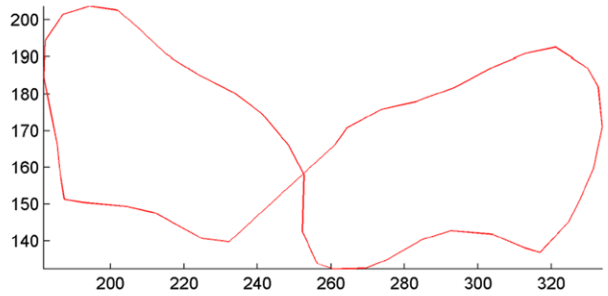
(c) Ellipse sequence. Noise=$2\delta_n$.



(d) Contours before splitting. Noise=$2\delta_n$.



(e) Ellipse sequence. Noise=$3\delta_n$.



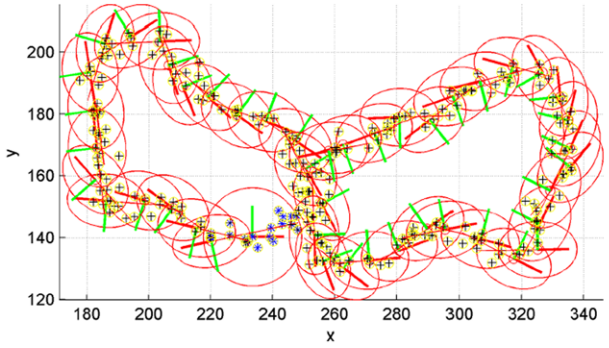
(f) Contours before splitting. Noise=$3\delta_n$.

Fig. 14 Algorithm performance. Slice data from *Pelvis* data set

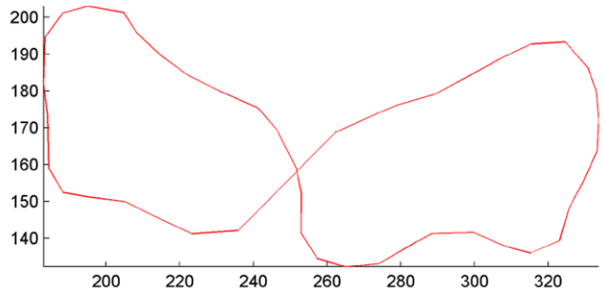
or geometrically erroneous. To get correct 1-manifolds we apply the already discussed post-processing for integrating the disconnected PL curves and the left- or right-splitting actions in Figs. 14(e) and 14(f), if needed.

4.3 Data set 3. Skull

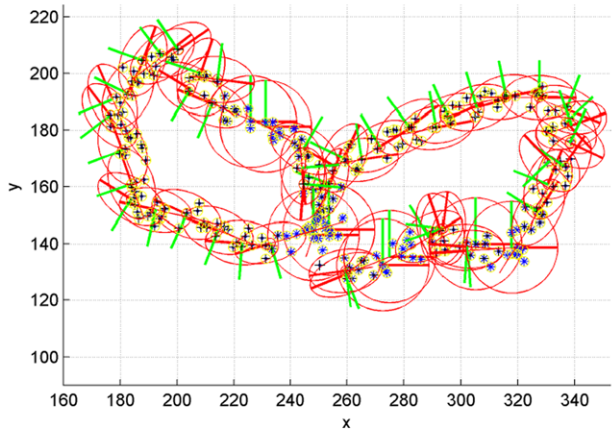
The *Skull* data set consists of 64 slices. Each slice contains nested and/or disconnected contours. Some levels have contours which are nearly self-intersecting, as seen in Fig. 15. A particular slice of such a data set contains a contour as the one shown in Fig. 15(a). Figures 15(b), 15(c) and 15(d) show point samples of the contour with sampling noise of $1\delta_n$, $3\delta_n$ and $6\delta_n$, respectively. It is evident that the point samples, even for low noise, reflect a nearly self-intersecting curve. Likewise, since the mentioned contours contain very fine geometric detail, the frequency content of them is quite high. As a consequence of the Nyquist principle, the minimal sampling distance needed to recover such contours is also very small (half of the size of the smallest geometric feature to be captured). This circumstance immediately reflects on the tightness of the sample, noise and the progres-



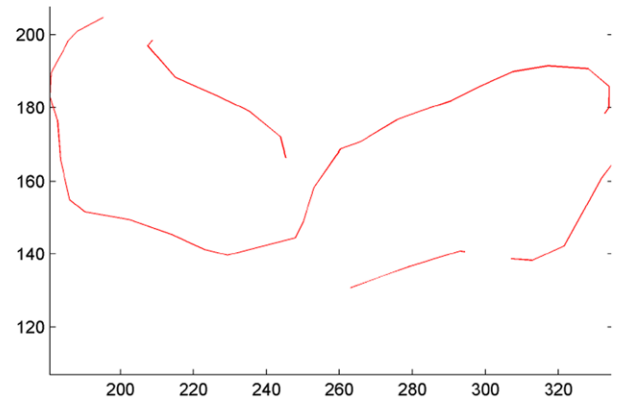
(g) Ellipse sequence. Noise=$4\delta_n$.



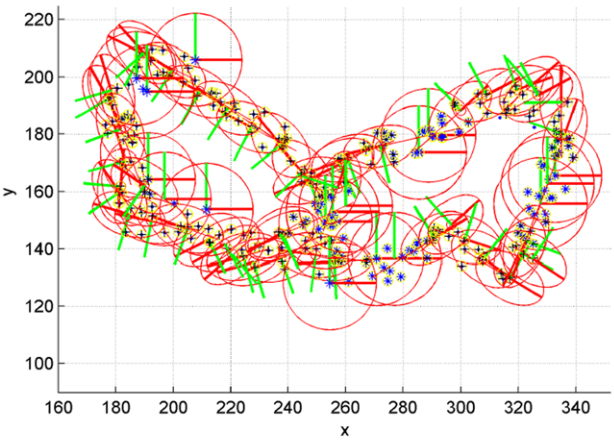
(h) Contours before splitting. Noise=$4\delta_n$.



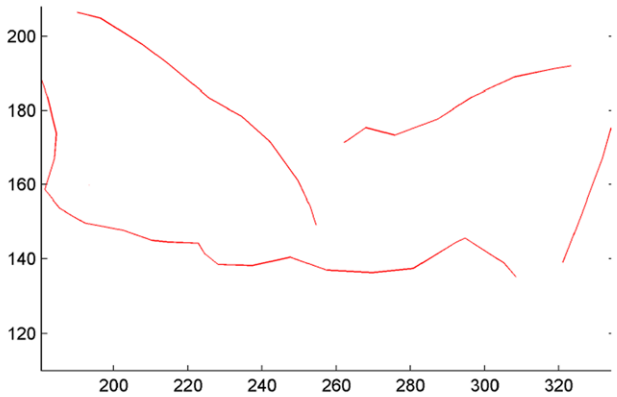
(i) Ellipse sequence. Noise=$5\delta_n$.



(j) Incomplete Contours. Noise=$5\delta_n$.



(k) Ellipse sequence. Noise=$6\delta_n$.

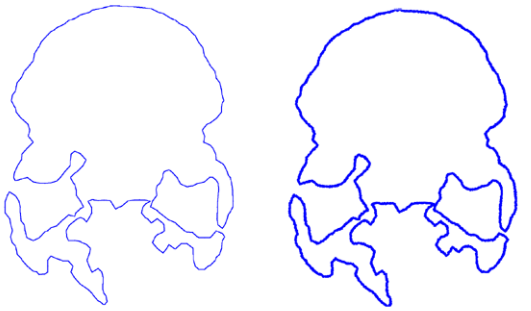


(l) Incomplete Contours. Noise=$6\delta_n$.

Fig. 14 (*Continued*)

sion of the ellipse evolution, all of them being very different as compared with the *Pelvis* data set.

Algorithm 1 is run using the data sets of Figs. 15(b), 15(c) and 15(d). The evolution of the ellipse algorithm for each noise level is displayed in Figs. 16(a), 16(c) and 16(e), respectively. The inherent difficulty in the contour processed produces a much tighter sequence of ellipses than the ones recorded in Fig. 14 (*Pelvis* data set). Figures 16(b), 16(d) and 16(f) illustrate the result of the execution of Algorithm 1. The results of the recovery of individual PL ap-



(a) A Contour on a Slice. (b) Sample with noise $1\delta_n$.

(c) Sample with noise $3\delta_n$. (d) Sample with noise $6\delta_n$.

Fig. 15 Noisy samples of a contour in the *Skull* data set

proximations of C from the random noisy point sets are satisfactory for the noise levels $1\delta_n$ and $3\delta_n$ but fail for noise level $6\delta_n$.

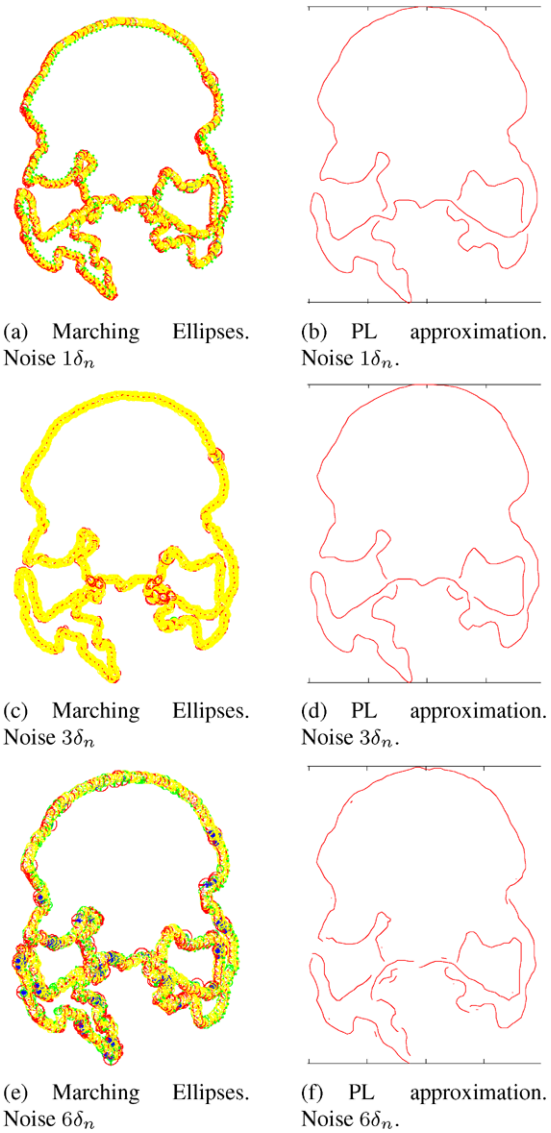
Notice that the individual PL curves are not exactly manifolds because they are self-intersecting. Moreover, they are still fragmented. Therefore, the individual PL curves are still to be appended together as in Figs. 14(b) and 14(c), and as discussed in section "Integration of PL Fragments." Next, the self-intersecting PL curves must be split at the self-intersections as shown in Fig. 14(d).

Figure 17(a) displays the *Skull* contour set as obtained by the iterated application of the algorithm discussed in the present article. Then, a surface reconstruction algorithm from parallel planar contours ([21] or [7]) was executed rendering the surface shown in Fig. 17(b).

4.4 Counting of elementary operations

Besides the discussion regarding the complexity of the implemented algorithms, the number of elementary operations were registered for the *Pelvis* and *Skull* runs. By *elementary operation* we mean the access to, or query of, any point by the algorithm. This register avoids working with execution times, which are not a reliable data, since they become meaningless after new hardware comes into market.

Figure 18 displays the result of the runs with the following conditions: (1) Two types of elementary operations recorded: (a) point accesses in ellipse inclusion tests, and (b) point access in point set decimation. (2) Noise levels $[1\delta_n, \ldots, 6\delta_n]$. (3) *Pelvis* and *Skull* data sets.



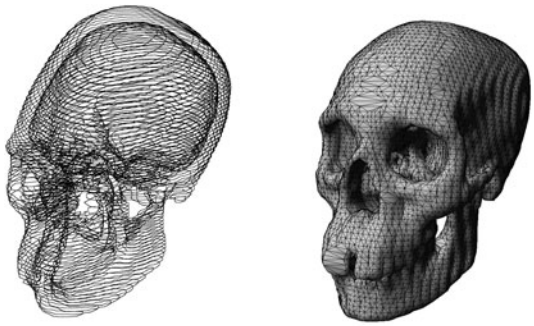
(a) Marching Ellipses. Noise $1\delta_n$ (b) PL approximation. Noise $1\delta_n$.

(c) Marching Ellipses. Noise $3\delta_n$ (d) PL approximation. Noise $3\delta_n$.

(e) Marching Ellipses. Noise $6\delta_n$ (f) PL approximation. Noise $6\delta_n$.

Fig. 16 Results. *Skull* data set

Figure 18(a) deals with the *Pelvis* data set. The upper curve represents the number of accesses to the points in point-in-ellipse queries. The lower curve shows the number of accesses to the points in point decimation operations. Decimations are performed to progressively reduce the search space (point set) after a separate PL curve is reconstructed. The decimation eliminates from the search space the points that have just participated in the estimation of the last PL curve. Fig. 18(b) corresponds to the *Skull* data set, with the upper and lower curves having the meaning just commented. It is evident from Fig. 18 that the *Skull* data set is considerably more complex that the *Pelvis* data set. We can also notice that the cost of both point access operations (i.e., point-ellipse inclusions and point decimation) is practically flat with respect to the level of noise of the data set, for both data sets.



(a) Recovered Contours. (b) Triangled Skull.

Fig. 17 Contours and reconstructed surface for the *Skull* data set

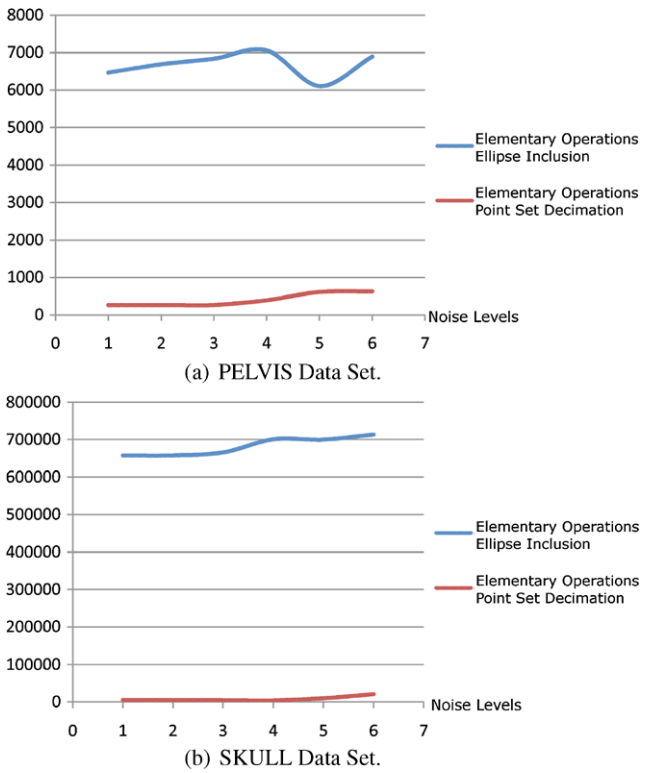


Fig. 18 Number of elementary operations (point accesses)

The fact that the point decimation is much cheaper than the point-ellipse inclusion is a somehow positive finding. It means that this collateral operation (decimation), which greatly expedites the runing of the main algorithm (point-ellipse inclusion), can be applied without significant additional burden.

5 Conclusions and future work

This article has presented an algorithm (test sets *Hand*, *Pelvis*, *Skull*) for the reconstruction of a planar curve C out of a noisy sample of it. The algorithm has the following characteristics: (1) It constructs a Piecewise Linear approximation of C. (2) It is able to recover self-intersecting or nearly self-intersecting curves rendering a decomposition of them into disjoint 1-manifolds. (3) It performs local Principal Component Analysis using support regions whose form mutates from circular disks, in neighborhoods where there are no self-intersections, to flat ellipses near the self-intersections. (4) It does not require collateral data structures or pre-processing, and its worst-case complexity is $O(N^4)$ where N is the number of points sampled on C.

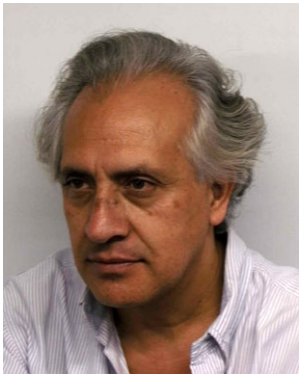
We consider this *worst*-case complexity as non-comparable with the complexity reported by some authors addressing the same problem, since they estimate *expected* cases and fail to account for the *computing time* and *space* spent in the collateral data structures and pre-processing present in their algorithms. It must be pointed out that the vast majority of the literature reviewed does not address computational expenses of their proposed algorithms.

Future work on the topic of curve reconstruction includes the reconstruction of non-planar curves, and the lowering of complexity of reconstruction with implicit forms of higher degree (Spline, Bézier, NURBS).

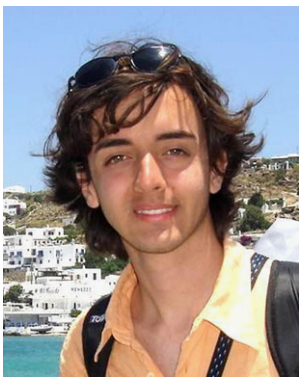
References

1. Arora, S., Khot, S.: Fitting algebraic curves to noisy data. J. Comput. Syst. Sci. **67**(2), 325–340 (2003)
2. Bloomenthal, J.: Polygonization of implicit surfaces. Comput. Aided Geom. Des. **5**(4), 341–355 (1988)
3. Bloomenthal, J.: An implicit surface polygonizer. In: Graphics Gems IV, pp. 324–349. Academic Press, San Diego (1994)
4. Bloomenthal, J., Ferguson, K.: Polygonization of non-manifold implicit surfaces. In: SIGGRAPH '95: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, New York, NY, USA, pp. 309–316. ACM Press, New York (1995)
5. Cheng, S.-W., Funke, S., Golin, M., Kumar, P., Poon, S.-H., Ramos, E.: Curve reconstruction from noisy samples. Comput. Geom. Theory Appl. **31**(1–2), 63–100 (2005)
6. Dey, T.K., Kumar, P.: A simple provable algorithm for curve reconstruction. In: SODA '99: Proceedings of the tenth Annual ACM-SIAM Symposium on Discrete Algorithms, Philadelphia, PA, USA, pp. 893–894. Society for Industrial and Applied Mathematics, Philadelphia (1999)
7. Geiger, B.: Three-dimensional modeling of human organs and its application to diagnosis and surgical planning. Technical Report RR-2105 (1993)
8. Kegl, B., Krzyzak, A.: Piecewise linear skeletonization using principal curves. IEEE Trans. Pattern Anal. Mach. Intell. **24**(1), 59–74 (2002)
9. Klein, J., Zachmann, G.: Point cloud surfaces using geometric proximity graphs. Comput. Graph. **28**(6), 839–850 (2004)
10. KTgl, B.: Principal curves: learning, design, and applications. PhD thesis, Concordia University, Montreal, Canada (1999)
11. Lee, I.K.: Curve reconstruction from unorganized points. Comput. Aided Geom. Des. **17**(2), 161–177 (2000)
12. Liu, Y., Yang, H., Wang, W.: Reconstructing b-spline curves from point clouds – a tangential flow approach using least squares minimization. In: SMI '05: Proceedings of the International Conference on Shape Modeling and Applications 2005, pp. 4–12, Washington, DC, USA. IEEE Computer Society, Los Alamitos (2005)

13. Lu, D., Zhao, H., Jiang, M., Zhou, S., Zhou, T.: A surface reconstruction method for highly noisy point clouds. In: Third International Workshop on Variational, Geometric and Level Set Methods in Computer Vision, VLSM, pp. 283–294 (2005)
14. Mukhopadhyay, A., Das, A.: Curve reconstruction in the presence of noise. In: CGIV '07: Proceedings of the Computer Graphics, Imaging and Visualisation, Washington, DC, USA, pp. 177–182. IEEE Computer Society, Los Alamitos (2007)
15. Niyogi, P., Smale, S., Weinberger, S.: Finding the homology of submanifolds with high confidence from random samples. Discrete Comput. Geom. **39**(1), 419–441 (2008)
16. Nyquist, H.: Certain topics in telegraph transmission theory. Bell Syst. Tech. J., **47** (1928)
17. Nyquist, H.: Certain topics in telegraph transmission theory. Proc. IEEE **90**(2), 617–644 (2002). Reprint as classic paper
18. Osher, S., Fedkiw R.: Level set methods: An overview and some recent results. Technical report, University of California Los Angeles, Stanford University (2000)
19. Osher, S., Sethian, J.A.: Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations. J. Comput. Phys. **79**(1), 12–49 (1988)
20. Pauly, M., Mitra, N.J., Guibas, L.: Uncertainty and variability in point cloud surface data. In: Alexa, M., Rusinkiewicz, S. (eds.) Symposium on Point-Based Graphics, pp. 77–84. Eurographics, Geneve (2004)
21. Ruiz, O., Cadavid, C., Granados, M., Peña, S., Vásquez, E.: 2D shape similarity as a complement for Voronoi–Delone methods in shape reconstruction. Elsevier J. Comput. Graph. **29**(1), 81–94 (2005)
22. Ruiz, O., Vanegas, C., Cadavid, C.: Principal component and Voronoi skeleton alternatives for curve reconstruction from noisy point sets. J. Eng. Des. **18**(5), 437–457 (2007)
23. Shannon, C.E.: Communication in presence of noise. Proc. IRE **37**(1), 10–21 (1949)
24. Shannon, C.E.: Communication in presence of noise. Proc. IEEE **86**(2), 447–457 (1998). Reprint as classic paper
25. Tagliasacchi, A., Zhang, H., Cohen-Or, D.: Curve skeleton extraction from incomplete point cloud. ACM Trans. Graph. **28**(3), 1–9 (2009)
26. Unnikrishnan, R., Lalonde, J.-F., Vandapel, N., Hebert, M.: Scale selection for the analysis of point-sampled curves. In: 3DPVT, pp. 1026–1033 (2006)
27. Uribe, D., Ruiz, O.: 2D curve reconstruction with heat transfer differential equations. Technical report, EAFIT University, CAD CAM CAE Laboratory, Nov. (2008)
28. Verbeek, J.J., Vlassis, N., Kröse, B.: A soft k-segments algorithm for principal curves. In: Proc. Int. Conf. on Artificial Neural Networks, pp. 450–456, Vienna, Austria, August (2001)
29. Wang, W., Pottmann, H., Liu, Y.: Fitting b-spline curves to point clouds by curvature-based squared distance minimization. ACM Trans. Graph. **25**(2), 214–238 (2006)
30. Zhao, H.-K., Osher, S., Fedkiw, R.: Fast surface reconstruction using the level set method. In: VLSM '01: Proceedings of the IEEE Workshop on Variational and Level Set Methods (VLSM'01), Washington, DC, USA, p. 194. IEEE Computer Society, Los Alamitos (2001)
31. Zhao, H.k., Oshery, S., Merrimany, B., Kangy, M., Implicit and non-parametric shape reconstruction from unorganized points using variational level set method. Comput. Vis. Image Underst. **80**, 295–319 (2000)



O. Ruiz (1961, Tunja, Colombia) obtained Diplomas in Mechanical Eng. and Computer Science at Los Andes University, Colombia, and M.Sc. and Ph.D. degrees with emphasis in CAD-CAM (1991, 1995) from University of Illinois at Urbana-Champaign, USA. Dr. Ruiz has been Visiting Researcher at Ford Motor Co. (USA), Fraunhofer Inst. for Computer Graphics (Germany), University of Vigo (Spain), Max Planck Institute for Informatik (Germany) and Purdue University (USA). Prof. Ruiz is the coordinator of the Laboratory of CAD / CAM / CAE at EAFIT University, COLOMBIA. His interests are Computer Aided Geometric Design and Applied Computational Geometry.



C. Vanegas (1984, Medellín, Colombia) is a Ph.D. student in the Department of Computer Science at Purdue University, USA, and a Research Assistant at the Computer Graphics and Visualization Lab under the supervision of Prof. Daniel Aliaga. He earned a diploma in Applied Mathematics from EAFIT University (2007), and was a Research Intern in the CAD/CAM/CAE Lab at EAFIT (2004–2007), University of Vigo, Spain (2005, 2006) and ETH Zurich (2010). His interests are in Computer Graphics and Computational Geometry, with focus on procedural methods for fast design and editing of 3D urban models using concurrent behavioral and geometrical simulation.

C. Cadavid (1965, Medellín, Colombia) obtained a Diploma in Mathematics at the Colombia national University (1988), a M.Sc. from the University of Cincinnati (1990), and a Ph.D. in Mathematics in 1998 from the University of Texas at Austin. He is currently a full time Faculty member at the Universidad EAFIT, COLOMBIA. His areas of research are the topology of 4-dimensional manifolds and applications of topology to Computer Aided Geometric Design.