Computers & Graphics (2018)

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Peeling the Longest: A Simple Generalized Curve Reconstruction Algorithm

ARTICLE INFO

Article history:
Received March 12, 2018

Keywords: Curve Reconstruction, Delaunay Triangulaton, Noise Simplification, Self-intersection

ABSTRACT

Given a planar point set sampled from a curve, the curve reconstruction problem computes a polygonal approximation of the curve. In this paper, we propose a Delaunay triangulation-based algorithm for curve reconstruction, which removes the longest edge of each triangle to result in a graph. Further, each vertex of the graph is checked for a degree constraint to compute simple closed/open curves. Assuming ϵ-sampling, we provide theoretical guarantee which ensures that a simple closed/open curve is a piecewise linear approximation of the original curve. Input point sets with outliers are handled as part of the algorithm, without pre-processing. We also propose strategies to identify the presence of noise and simplify a noisy point set, identify self-intersections and enhance our algorithm to reconstruct such point sets. Perhaps, this is the first algorithm to identify the presence of noise in a point set. Our algorithm is able to detect closed/open curves, disconnected components, multiple holes and sharp corners. The algorithm is simple to implement, independent of the type of input, non-feature specific and hence it is a generalized one. We have performed extensive comparative studies to demonstrate that our method is comparable or better than other existing methods. Limitations of our approach have also been discussed.

1. Introduction

Curve reconstruction of a given set of points S, sampled from a curve C, computes a polygonal (piecewise linear) approximation of the curve. The curve C can be an open or a closed curve with self-intersections, disconnected components, multiple holes and sharp corners, where a hole (inner boundary) is considered as a convex/non-convex simple polygon which is enclosed within a boundary. The input and output of the problem are as shown in Figures 1(a) and 1(b), respectively. Even though the reconstruction problem, in general, has a rich literature over the last three decades, it is still an active and challenging problem due to its ill-posed nature [1]. Nevertheless, it has various applications in the fields of computational geometry, computer vision, computer graphics, image processing and pattern recogonition.

Edelsbrunner proposed a Delaunay triangulation-based parametric method to produce α-shape [2], which characterizes the shape of a point set. Even though it was not designed for curve reconstruction, subsequently its 3D version [3] was shown to be

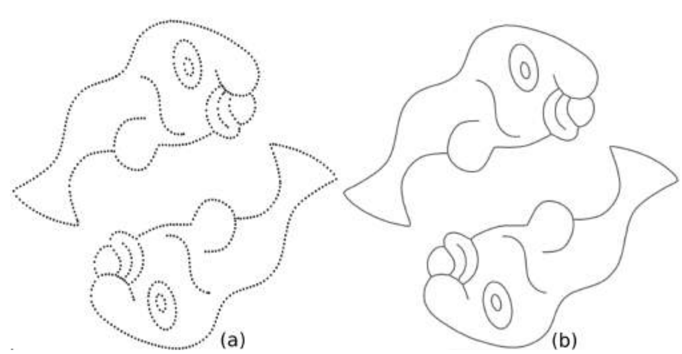


Fig. 1: (a) Input point set. (b) Our output with closed/open curve, disconnected components, self-intersections, multiple holes and sharp corners.

applicable for reconstruction. $\mathcal{A}$-shape [4] is computed from a combination of Delaunay triangulation and Voronoi diagram. Veltkamp proposed a method for reconstruction which results in γ-graph [5]. Duckham et al. [6] proposed a parametric method for reconstruction from the Delaunay triangulation to produce characteristic shape (χ-shape). Locally Density Adaptive α-complex [7] is a Delaunay triangulation-based, locally adaptive method for curve reconstruction. Simple shape [8] computes the curve, based on the distance, angle and certain other criteria. Concepts of Delaunay disks are used to compute r-regular shape [9]. Crust algorithms [10, 11, 12] use a combination of Delaunay triangulation and Voronoi diagram to produce closed/open curves. In Crust, a dense sampling based on medial axis transform was introduced by Amenta et al. [10], which is widely used to ensure theoretical guarantee of a reconstructed curve. Reconstruction using nearest neighbour graph with theoretical guarantee is presented in nn-crust [11]. In power crust [12], a subset of Voronoi vertices known as poles is used to build a power diagram, which divides the plane into interior and exterior cells. Noise filtering of a given point set and introducing new points, followed by pruning and reconstruction using nn-crust is proposed by Cheng et al. [13]. Mehra et al. [14] proposed a visibility operator on the convex hull of a noisy point set and in turn used the visibility information to perform both curve and surface reconstructions. Feiszli et al. [15] introduced a non-parametric denoising strategy and reconstructing a curve preserving sharp corners. However, the three curve reconstruction algorithms mentioned above [13, 14, 15] do not reconstruct open curves, disconnected components, curves with self intersections and they are not designed for handling outliers. Lee [16] proposed a reconstruction method based on moving least squares concept, specially designed for noisy point sets to compute curves without self-intersections. Shape hull [17] removes the edges of a Delaunay triangulation based on the position of circumcenter of triangles to construct a simple closed divergent curve. Another Delaunay triangulation-based method is ec-shape, which use empty circle approach for outer boundary detection [18] and hole detection [19]. Non-divergent curves are also reconstructed by ec-shape, but not open curves. Water-distribution-model (wdm) crust [20] is based on Voronoi diagram and handles outliers. Crawl [21] reconstructs closed/open curves with disconnected components and multiple holes, however it does not handle noisy point set. Optimal transport cost method proposed by deGoes et al. [22] is a greedy method to minimize the increase in the transport cost, which is designed for noisy point sets. Wang et al. [23] proposed a quad-tree method with smoothening concept to reconstruct a curve from a noisy point set with outliers. There are algorithms such as Fidelity vs. Simplicity [24], which perform a piecewise smooth reconstruction of a given sketch.

Most among the above methods are designed only for a simple closed curve reconstruction [6, 17, 18] whereas a few of them [21, 22, 23] reconstruct both open and closed curves. To the best of our knowledge, only few of them [10, 11, 20] provide theoretical guarantee. Apart from reconstructing open and closed curves, few algorithms are designed to handle other challenges of reconstruction problem such as (i) the input can be noisy or/and with outliers (ii) the original curve can have disconnected components, self-intersections, multiple holes and sharp corners. Even though a few of the reconstruction algorithms [18, 20, 21] detect some of the features mentioned above, they are not designed for handling noise. There are algorithms specially designed for handling noise [13, 14, 15], but they do not reconstruct open curves, disconnected components, curves with self intersections and are not designed for handling outliers. Lee [16] designed a reconstruction algorithm for noisy point sets for both closed and open curves, however, it is not able to detect self-intersections. F. de Goes et al. [22] and Wang et al. [23] detect self-intersections on a noisy point set, however performance of Wang et al. [23] degrades if the input is without noise. Also, how to detect the presence of noise remains to be a challenging open problem. Moreover, many algorithms have multiple parameters, hence it is very tedious to synchronize and tune.

Summary of Comparison							
Method	OC	DC	SI	NI	IO	NFS	NP
α-shape [2]	*Y*	*Y*	*Y*	*N*	*N*	*Y*	1
χ-shape [6]	*N*	*N*	*N*	*N*	*N*	*Y*	1
Simple shape [8]	*N*	*N*	*N*	*N*	*N*	*Y*	3
Crust [10]	*Y*	*Y*	Y^*	N^*	Y^*	*Y*	0
nn-crust [11]	*Y*	*Y*	Y^*	N^*	Y^*	*N*	0
Cheng et al. [13]	*N*	*N*	*N*	*Y*	*N*	*N*	3
Mehra et al. [14]	*N*	*N*	*N*	*Y*	*N*	*N*	2
Feiszli et al. [15]	*N*	*N*	*N*	*Y*	*N*	*N*	0
Lee [16]	*Y*	*Y*	*N*	*Y*	*Y*	*Y*	0
Shape hull [17]	*N*	*N*	*N*	*N*	*N*	*N*	0
ec-shape [18, 19]	*N*	*N*	*N*	*N*	*N*	*Y*	1
wdm-crust [20]	*N*	*N*	*N*	*N*	*Y*	*Y*	0
Crawl [21]	*Y*	*Y*	*N*	*N*	*Y*	*Y*	0
deGoes et al. [22]	*Y*	*Y*	*Y*	*Y*	*Y*	*N*	2
Wang et al. [23]	*Y*	*Y*	*Y*	*Y*	*Y*	*N*	4
Our Method	*Y*	*Y*	*Y*	*Y*	*Y*	*Y*	$2^@$

Table 1: Summary of comparison of existing methods, where *Y* or *N* refers to YES or NO based on whether the corresponding method handles the following: Open Curve (OC), Disconnected Components (DC), Self-Intersections (SI), Noisy Input (NI), Input with Outliers (IO), Non-Feature-Specific (NFS) and Number of Parameters (NP). *Crust and nn-crust are not specifically designed for reconstructing self-intersections, noisy inputs and outliers, hence, the reconstructed output may or may not handle these. @ One parameter for simplifying the noisy point set and another one for identifying the presence of self-intersections.

Table 1 summarizes the comparison of the existing methods, where *Y* or *N* refers to YES or NO based on whether the corresponding method handles the following: Open Curve (OC), Disconnected Components (DC), Self-Intersections (SI), Noisy Input (NI), Input with Outliers (IO), Non-Feature-Specific (NFS) and Number of Parameters (NP).

In general, it is challenging to develop a generalized algorithm which handles all the features - closed/ open curves, disconnected components, self-intersections, multiple holes and sharp corners as well as handle noise and outliers.

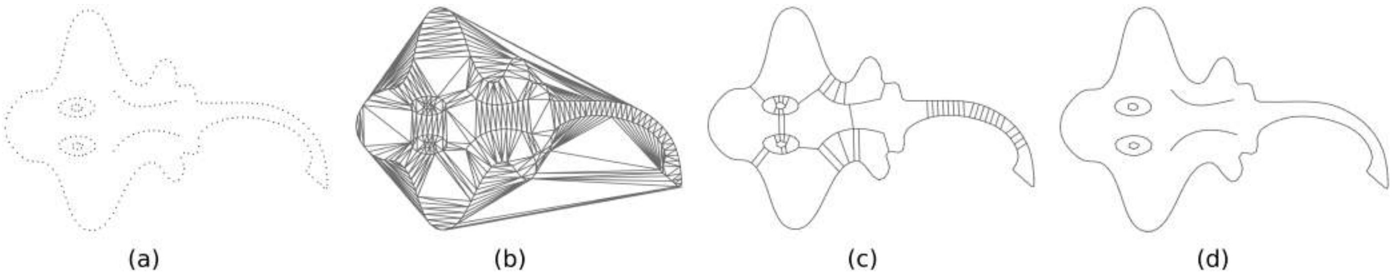


Fig. 2: Illustration of Algorithm 1 *(a)* Input point set *(b)* Delaunay triangulation *(c)* Intermediate output after removing the longest edge from all of the triangles *(d)* The final output after checking the degree constraint for every point. Output has features such as open and closed curves, multiple holes and sharp corner.

1.1. Our contributions

We have developed an algorithm which reconstructs simple closed/open curves with theoretical guarantee. Point sets with outliers are also handled by this algorithm without preprocessing. Further, we have designed strategies to identify the presence of noise and self-intersections. Our algorithm is enhanced to simplify a noisy point set and perform curve reconstruction. It is also extended to reconstruct curves with self-intersections. The algorithm is simple to implement too. Hence, our algorithm is a generalized one as it is not designed for a particular input case or feature-specific, but tuned to handle different input cases and detects various features.

Our major contributions are as follows:

- Non-feature specific algorithm for reconstruction of closed/open curves.
- Novel 'flower structure' to identify the presence of noise (perhaps for the first time), using Delaunay triangulation.
- Strategy to identify and restore self-intersections in the reconstructed curve.

Algorithm 1: Reconstruct(S)

Input: Input point set S
Output: Simple curve(s)
Construct Delaunay Tringulation (DT) of S
for *each triangle T in DT* **do**
  Remove the longest edge
end
Let G be a graph representing the remaining edges in DT
for *each point p in G, which does not satisfy degree constraint* **do**
  repeat
    Remove the longer edge incident at p
  until *degree constraint of p is satisfied*;
end
return G

2. Algorithm

Let DT denote the Delaunay triangulation [25] of an input point set S. An edge between a pair of points (vertices) in DT is denoted by e. It has been proved that, if S is obtained by ϵ-sampling [10] from a simple closed curve, a set of connected subgraphs of DT provides a piecewise linear approximation of S [10]. Hence, we use this fact to develop our reconstruction algorithm, based on Delaunay triangulation.

Algorithm 1 starts with the construction of a Delaunay triangulation DT of the input point set S. From each triangle T of the Delaunay triangulation, the longest edge e is removed, which results in a graph G. In G, for each point p with more than two as degree, at most two shortest incident edges are retained (this condition of having degree at most two for all the points in the reconstructed curve is known as degree constraint). Pseudocode of our approach for reconstructing a closed/open curve is given in Algorithm 1.

Figure 2 illustrates various steps of Algorithm 1. Figure 2(a) and Figure 2(b) show the input point set and its DT respectively. The result after removing the longest edges from all the triangles is shown in Figure 2(c). The final output after checking the degree constraint for every point is shown in Figure 2(d). Our result captures features such as closed and open curves, multiple holes and sharp corners. Please note that Algorithm 1 which reconstructs simple closed/open curves, is a non-parametric one.

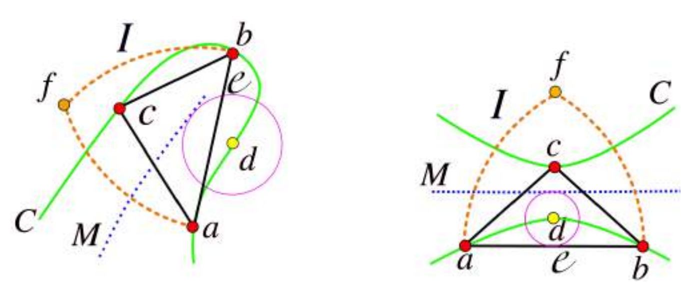


Fig. 3: Two cases where a longest edge becomes the part of a piecewise linear approximation of the original curve C.

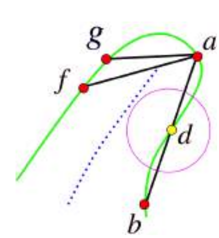


Fig. 4: A case in which the edge in a piecewise linear approximation of original curve C is removed due to degree constraint.

2.1. Guaranteeing the results

Let R be the reconstructed curve using our algorithm and P be a piecewise linear approximation of a curve C. Theoretical guarantee implies that the output of Algorithm 1 is a piecewise linear approximation of the ϵ-sampled curve C.

DEFINITION 1 Medial axis of a curve C is a closure of the set of points in the plane which has two or more closest points in C [10].

DEFINITION 2 Local feature size $LFS(p)$ of a point $p \in C$ is the Euclidean distance from p to the closest point m on the medial axis [10].

DEFINITION 3 A point set is an ϵ-sampling of a curve C, if for every $p \in C$, there is a sample within a distance $\epsilon \times LFS(p)$, where $0 < \epsilon < 1$ [10].

In the reconstructed curve, if there is an edge existing between a pair of points, they are known to be adjacent to each other, if not, they are non-adjacent.

2.1.1. Closed Curve

To ensure theoretical guarantee of our algorithm, it is enough to prove that an edge $e \in P \Leftrightarrow e \in R$, where R is the reconstructed curve using our algorithm and P is a piecewise linear approximation of the original simple closed curve C.

LEMMA 2.1 $e \in P \Rightarrow e \in R$

Proof Suppose $e \in P$ but $e \notin R$, e might have removed because of the following reasons:

- Case 1: e is the longest edge in one of the Delaunay triangles
- Case 2: one of the endpoints of e has degree more than two

Case 1: Let a and b be the end points of e, and c be the third point in the Delaunay triangle in which e is the longest edge. Figure 3 shows illustrations of this case, where curve(s) C and corresponding medial axis M are shown in green and blue (dotted lines) color respectively. As C is a simple closed curve, a, b and c are not mutually adjacent to each other in P. Let I be the region defined by half of the intersection of circles drawn with a and b as centers and $||ab||$ as radius (yellow dotted arcs in Figure 3). The point c should lie inside I since $||ab|| > ||ac||$ and $||ab|| > ||bc||$. If $ac \notin P$, then the vertices a and c are on the different sides of the medial axis. On the other hand, if $bc \notin P$, the vertices b and c are on the different sides of the medial axis. In both the cases, irrespective of the position of c inside I, there exists a point d in the curve segment of C between a and b for which the $\epsilon \times LFS(d)$ is empty (shown in magenta color of Figure 3) and hence contradicts the assumption that points of R follows ϵ-sampling of C. Absence of such a point d violates the fact that c should be inside I.

Case 2: Let a and b be the end points of e, in this case, either a or b has degree more than two (Figure 4 shows a sample case). Let b is connected to two other points f and g, which lies in a distance lower than $||ab||$ from b. Since either bf or bg is part of P, the other vertex (which is not adjacent to b in P) and a lies on the different sides of the medial axis. This makes a point d in the curve segment of C between a and b to lie closer to medial axis than to a and b, which makes point set of R not to fulfill ϵ-sampling condition. □

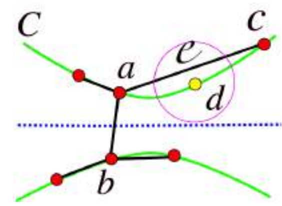


Fig. 5: A sample case in which an edge which is not present in a piecewise linear approximation of original curve C is present in the reconstructed output.

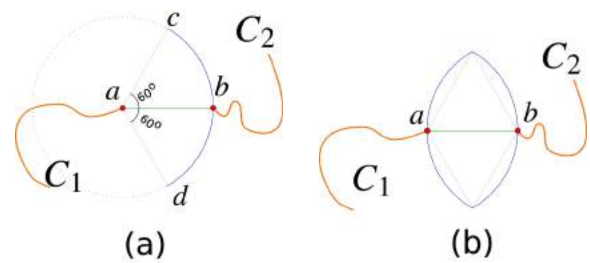


Fig. 6: C_1 and C_2 are parts of an original curve. a and b are end points of C_1 and C_2. (a) abc and abd are prominent sectors of a. (b) Prominent region of ab is the intersection of prominent sectors of a and b.

LEMMA 2.2 $e \in \mathrm{R} \Rightarrow e \in P$

Proof By Lemma 2.1, $e \in P \implies e \in R$, which means all the edges in a piecewise linear approximation of curve C is present in R. Suppose $e \in R$ and $e \notin P$, since C is a simple closed curve and all the edges in P are already present in R, the end points of e have degree 3 in R. It is enough to prove that e is the longest edge connected to end points of e and hence is removed due to the degree constraint. Let $e = ab$, and c be the adjacent point to a (or b) for which $||ab|| < ||ac||$. This makes a point d in the curve segment of C between a and c to have $\epsilon \times LFS(d)$ disk empty and hence contradicts our assumption that point set in R follows an ϵ-sampling of C. This case is shown in Figure 5. □

From Lemma 2.1 and Lemma 2.2, it follows that our algorithm reconstructs a piecewise linear approximation of the simple closed curve C.

2.1.2. Open Curve

Let C_1 and C_2 be two original curves and a and b are be one of the end points of C_1 and C_2 respectively.

DEFINITION 4 Prominent sectors of a point a are two sectors of a circle with radius $||ab||$ and center as a with center angle as 60° with respect to the edge ab ie. $\angle bac = \angle bad = 60°$.

Prominent sectors of a are abc and abd, as shown in Figure 6(a). Note that if the angle is greater than 60°, ab will not be the longest edge in the $\triangle abc$, where $\triangle abc$ is one of the Delaunay triangles.

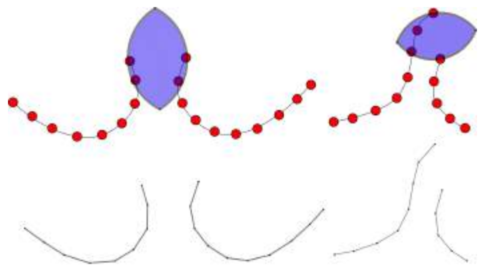


Fig. 7: First row shows a pair of input curves. Their prominent regions are non-empty and hence open curves are reconstructed as shown in second row.

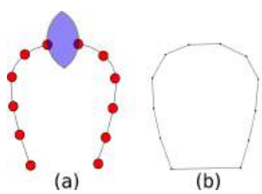


Fig. 8: (a) Input where the prominent region is empty (b) An invalid case in the proof where a closed curve is reconstructed instead of an open curve.

DEFINITION 5 Let a and b be two end points of the original curve C_1 and C_2. Prominent region with respect to edge ab is the intersection of prominent sectors of a and b (as shown in Figure 6(b)).

LEMMA 2.3 Let a and b be the end points of a curve/curves. The edge ab is removed if its prominent region is non-empty.

Proof If the prominent region of edge ab is non-empty (let point c be present in the region), then there exists a $\triangle abc$ in the Delaunay triangulation of the input point set. Since c is inside the prominent region, $\|ab\| > \|ac\|$ and $\|bc\|$ and ab will be removed by our algorithm. □

First row of Figure 7 shows input curves where respective prominent regions (which is shaded) are non-empty and the second row of shows reconstructed output curves. Figure 8 shows empty prominent region (which is shaded) and the reconstructed output.
If a prominent region of an edge connecting the end points is non-empty, the edge between the end points of the curves is removed resulting in an open curve.

THEOREM 2.4 Let the input point set be sampled from a set of curves (for which prominent regions of edges connecting the end points of open curves are non-empty), our algorithm reconstructs a piecewise linear approximation of the original curves.

Proof As a corollary to Lemma 2.1, 2.2 and 2.3, the shape of the original curve is captured and the end points have degree one, resulting in open curve(s). □

2.2. Handling curves with self-intersections

Self-intersection is a common feature in many planar curves. However, only a few of the existing methods [22, 23] can reconstruct self-intersections. Crust [10] might reconstruct curves with self-intersections for some point sets, since the output is a set of edges that do not have a restriction on the degree of vertices. Other existing algorithms are not generalized enough to work equally well in the presence and absence of noise.

We propose a strategy to detect the presence of self-intersections of a given point set. This is motivated by the snapping procedure available in authoring tools such as Adobe animate CC.

To check the presence of a self-intersection and to restore it in the reconstructed output, the following procedure is used: Reconstruct the input point set using Algorithm 1, which results in curves without self-intersections. Consider the reconstructed curve and DT of the input point set. Take any point p from the DT. Let d be the distance of the farthest incident point of p and ϑ be a user given parameter. Existence of another point k in the disc of radius $d\times\vartheta$ centered on p suggests the presence of a self-intersection. Adding a DT edge between p and k (if present) in the reconstructed curve (of Algorithm 1) restores the self-intersections. This strategy, instead of applying at all the points on the reconstructed curve, we employ only on the one degree vertices (as self-intersections are most likely to happen there). Self-intersections are restored with respect to the existence of other points in the disc of radius $d \times \vartheta$. Please note that our algorithm is capable of restoring multiple self-intersections with respect to a particular point.

Figure 9 shows an example of detecting and restoring a self-intersection. Figure 9(a) is the input point set and Figure 9(b) shows the reconstructed curves from Algorithm 1. Enlarged part of Figure 9(b) shows that self-intersection is not detected with Algorithm 1. Figure 9(c) is the Delaunay triangulation of the input point set and its enlarged portion shows the part of Delaunay triangulation where a self-intersection exists. Let the disc is with radius $\|pq\|$ (longest edge among pq and pr) and center p. Dilating the disc based on ϑ makes s to lie inside the disc, hence ps is added to Figure 9(b) thus introducing a self-intersection in Figure 9(d).

Figure 10 shows the effect of parameter tuning while restoring self-intersections. Parts which do not detect self-intersections well are shown in dotted circles. The parameter ϑ as 1.4 provides a better curve and even though we increase the value to 1.6, the curve remains unchanged since all open edges are paired with some other points in the reconstructed curve. From various experiments, we have observed that the value of ϑ varies from 1 to 2.

2.3. Simplifying a point set to remove the effect of noise

Curve reconstruction algorithms [13, 14, 15] designed for noisy point sets, do not reconstruct open curves, disconnected components, curves with self intersections. They do not handle outliers also. Curve reconstruction methods for noisy inputs [16, 22, 23] detect open curves and disconnected components. The method proposed by Lee [16] does not detect self-intersections and the method by Wang et al. [23] degrade its performance for a point set without noise. Those methods also do not detect the presence of noise a priori, rather assume that input contains noise.

We propose an approach (which is perhaps the first approach) to detect the presence of noise in an input point set S. We define

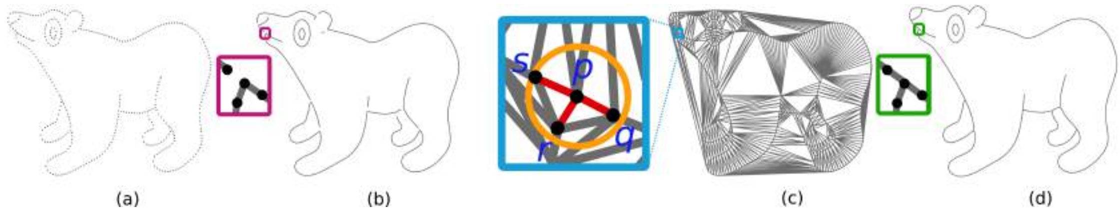


Fig. 9: *(a)* Point set. *(b)* Output of Algorithm 1 without restoring self-intersection (shown as enlarged) *(c) DT* with a part of triangulation (shown as enlarged) where p is the point under consideration, q is the far apart point incident on p, disc with radius $\|pq\| * \vartheta$ encloses s, *(d)* Our result after restoring self-intersection.

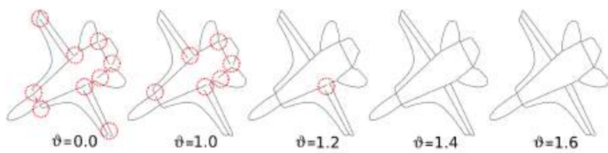


Fig. 10: Tuning of parameter to detect self-intersections. Parts in which self-intersections are not detected are shown in dotted circles. The best approximation is achieved at ϑ =1.4. The output remains same for all $\vartheta > 1.4$.

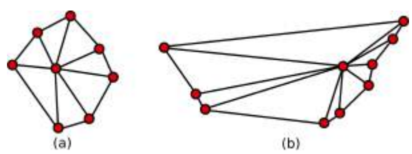


Fig. 11: (a) A valid flower structure (b) Invalid flower structure

a valid flower structure in the Delaunay triangulation of S. Let v be a vertex with a set of incident edges L. Let l_1 be the shortest edge and l_2 be the longest edge from L. A valid flower structure is a set of triangles associated with v, in which the ratio of l_2 to l_1 is lower than 2. This is based on the observation that for a non-noisy curve following ϵ-sampling, the ratio of longest incident edge to the shortest incident edge should be lower than 2.

Figure 11(a) shows an example of a valid flower structure and Figure 11(b) shows an example of an invalid flower structure. The existence of a valid flower structure as shown in Figure 12 with respect to a point p suggests the presence of noise. The noisy point set (Figure 12(a)) is simplified by removing all the points inside the disc with center p and radius μ, where μ is a user given parameter. The valid flower structure (enlarged part in the blue box of Figure 12(b)) ensures that the point p lies completely inside the boundary defined by the noisy point set and hence it is retained. After simplifying the point set (Figure 12(c)), we apply Algorithm 1 on the retained points to reconstruct the curve (Figure 12(d)).

Figure 13 shows the effect of parameter tuning while detecting the presence of noise and simplifying the point set. The parameter μ as 90 detects the presence of noise and simplifies the point set in a better way. The overall reconstruction procedure is shown in Algorithm 2, which has a worst case time complexity of $O(n^2)$.

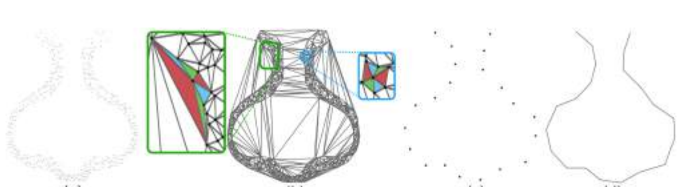


Fig. 12: (a) Noisy input point set ([16]) (b) Valid flower structure shown in enlarged blue box and Invalid flower structure shown in enlarged green box (c) Point set after simplification (d) Output of our algorithm

Algorithm 2: Complete_Reconstruction(S)

Input: Input point set S
Output: Reconstructed curve(s)
Construct Delaunay Tringulation (DT) of S
Search for a valid flower structure in $DT(S)$
if *a valid flower structure is present* **then**
  Read value for μ from user and initialize $\bar{S} = \phi$
  for *each vertex p having a valid flower structure* **do**
    Remove all points q from S, if $\|pq\| < \mu$
    $\bar{S} = \bar{S} \bigcup P$
  end
  $S = \bar{S}$
end
C=*Reconstruct*(S)
if *C has self-intersection(s)* **then**
  Read ϑ from user
  for *each vertex p in C with degree 1* **do**
    Find all points q in one-ring neighborhood of p in $DT(S)$ such that $\|pq\| < \vartheta \times l$, where l is the smallest edge length in the one-ring neighborhood of p
    Create edges from all q to p and add it to C
  end
end
return G

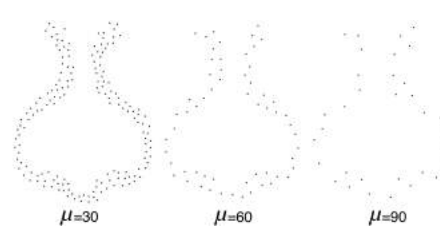


Fig. 13: Effect of tuning parameter for detecting the presence of noise and simplifying the point set. μ = 90 simplified the point set in a better way.

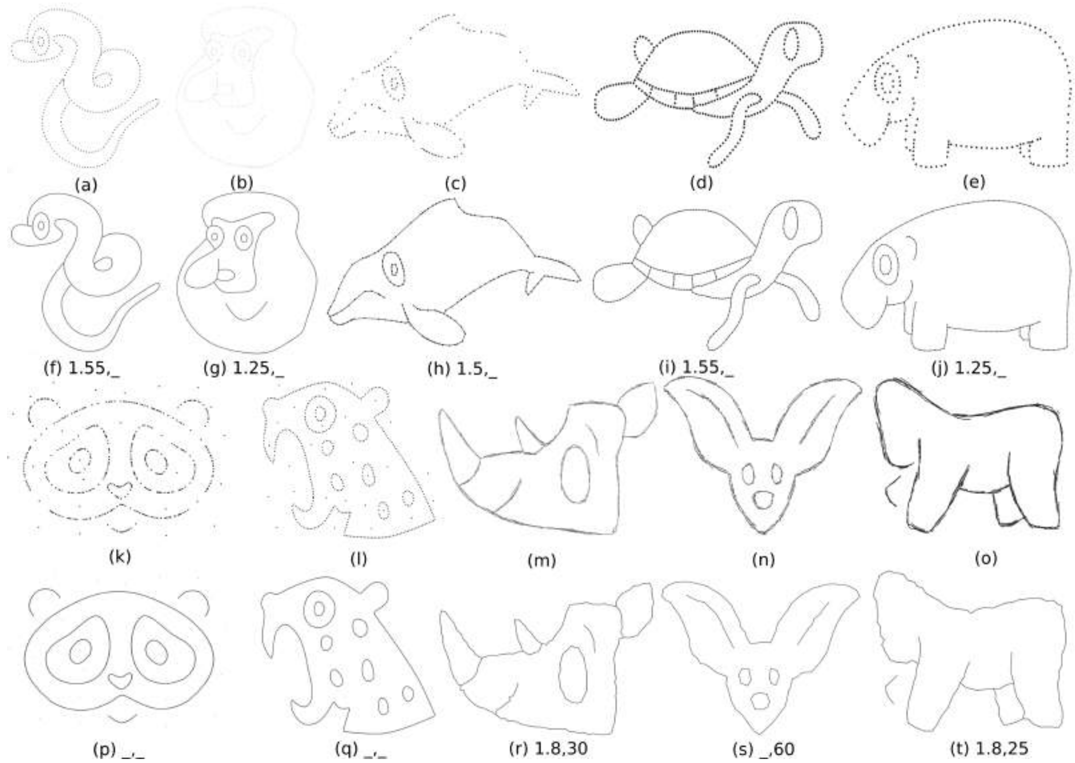


Fig. 14: (a)-(e) Input point set without noise and outliers, (f)-(j) Outputs of our algorithm with various features: (f) and (g) Self-intersections, multiple holes, elongated parts, (h) Open curves, multiple holes, sharp corners, self-intersections, (i) Concave parts, elongated features, self-intersections, (j) Concave parts, open curves, multiple holes, (k)-(l) Input point set with outliers, (m)-(o) Input point set with noise, (p)-(q) Reconstructed curve for inputs with outliers, (r)-(t) Reconstructed curve after simplification of noisy point set. The parameters (ϑ and μ for restoring self-intersection and noise simplification) are specified along with each result. An underscore is used to specify that no parameter is used.

3. Results and Discusssion

The algorithms are implemented in C++ using CGAL [26] and results are visualized using OpenGL framework. The point sets are sampled from inputs having open curves, self-intersections, sharp corners, disconnected components and multiple holes. The sampling is done using WebPlotDigitizer tool. The point set is sampled with various densities and random distributions along the curve. Please note that we do not follow any sampling assumptions for practical purposes, even though we assumed ϵ-sampling for ensuring theoretical guarantee (Section 2.1).

Figures 14(a)-14(e) show input point sets without noise and outliers. Different features such as self-intersections (e.g.: Figures 14(f), 14(g), 14(h), 14(i), 14(j)), sharp corners (Figure 14(h)), elongated features (e.g.: Figure 14(f), 14(i)), open curves (e.g.: Figures 14(g), 14(h), 14(j)), multiple holes (e.g.: Figures 14(f), 14(g), 14(h), 14(j)), concave parts (e.g.: Figure 14(i), 14(j)) are reconstructed.

Figures 14(k)-14(l) and Figures 14(m)-14(o) show input point sets with outliers and noise respectively. Figures 14(p)-14(q) show the reconstructed curves for inputs with outliers and Figures 14(r)-14(t) show the reconstructed curves after simplification of noise. Please note that the inputs are of different point densities (e.g.: Figures 14(a) and 14(e)) and of random distribution (e.g.: Figure 14(c)). The parameters (ϑ and μ for restoring self-intersection and simplification of noisy point set) are specified along with each result. If there is no parameter used, that is represented using an underscore.

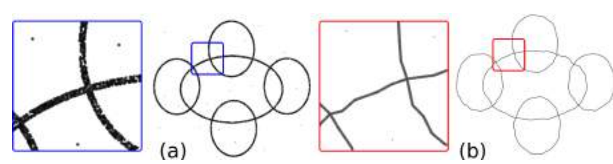
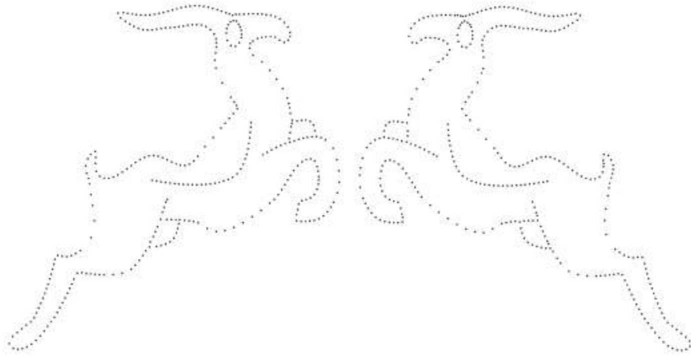


Fig. 15: (a) Input point set with noise, outliers and self intersections (b) Output of our algorithm after the removal of outliers, simplification of the point set and restoration of self-intersections. The parameters are $\vartheta = 1.7$ and $\mu = 20$ for restoring self-intersection and noise simplification respectively.

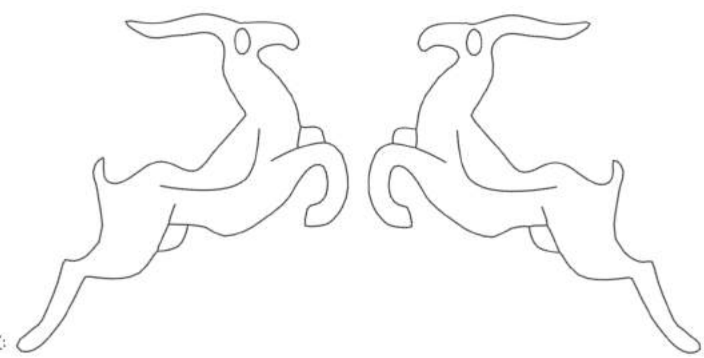
Figure 15(a) shows an input point set with outliers, noise and self-intersections and Figure 15(b) shows the output of Algorithm 2. Note that the outliers are removed, the point set is simplified and the self-intersections are restored.

3.1. Comparison with other approaches

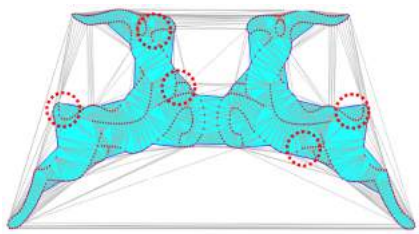
To evaluate the strength of our algorithm, we compared our results with those of existing reconstruction algorithms (α-shape [2], Crust [10], nn-crust [11], χ-shape [6], Simple shape [8], F. de Goes et al. [22], ec-shape [18, 19], wdm-crust [20] and Crawl [21]). Figure 16 shows the results generated by various algorithms for an input point set (without noise and outliers) of antelope with various features. For parametric methods, we have taken the best result obtained after parameter tuning. The parameters used are specified along with each



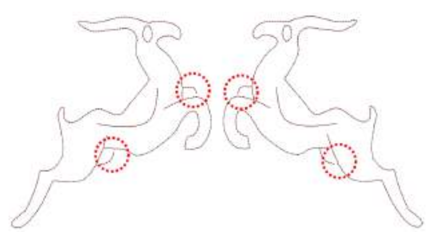
(a) Input



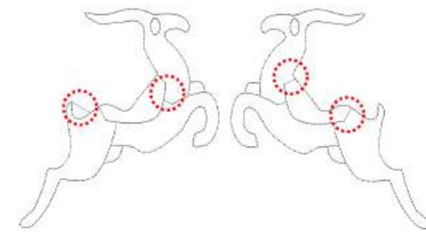
(b) Our result (ϑ=1.6)



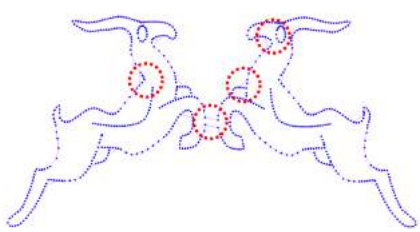
(c) α-shape (0.85)



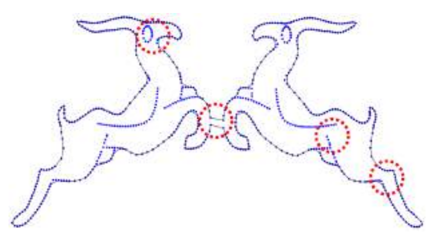
(d) Crust



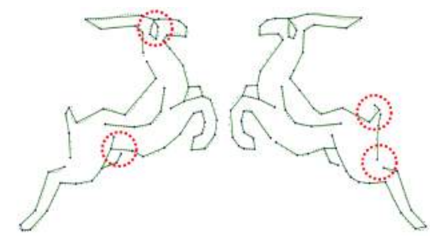
(e) nn-crust



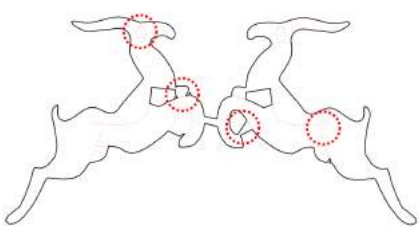
(f) χ-shape (3)



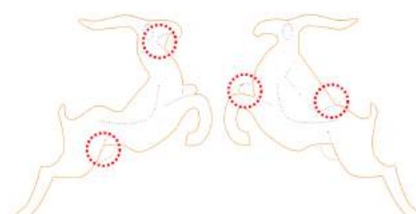
(g) Simple-shape (6,5,1)



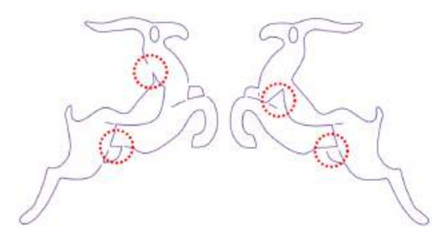
(h) F. de Goes et al. (136*)



(i) ec-shape



(j) wdm-crust



(k) Crawl

Fig. 16: Comparison with other algorithms: Our algorithm captures self-intersections better than other algorithms (the parts which are not detected well are shown in red dotted circles.). Disconnected components are not detected well by α-shape, χ-shape, Simple shape and ec-shape. Holes are detected by our algorithm better than α-shape, F. de Goes et al., ec-shape and wdm-crust. The parameters used are specified along with each result for the parametric methods in brackets.

result in brackets. For example, our method uses $\vartheta = 1.6$ for restoring self-intersections.

α-shape is not able to capture concave parts, holes and disconnected components properly. Crust was able to capture the curve comparatively well, but could not capture all self-intersections. nn-crust is not able to capture the open curves properly. As χ-shape and Simple shape are designed to reconstruct a closed simple polygon, they are not able to capture disconnected components. A few concave parts are also not captured well by them. Optimal transport approach by F. de Goes et al. have a few broken curves. As the algorithm is designed for noisy point set, it over simplified the curve which led to the loss of geometry. ec-shape is designed only for simple closed curve and it could not detect disconnected components. EC-shape could not detect the eyes of antelope as a hole. Even though wdm-crust detected an incomplete outer boundary, since it is not designed for objects with holes, it could not detect features lying inside the antelope curve. Constraints put on Crawl made it to ignore self-intersections and create a few non-appropriate edges in the curve. Our algorithm is able to capture the curve by preserving all the features.

Figure 17 shows the comparison of our algorithm with other algorithms based on the ability to capture various features. The parameters used are specified along with each output. α-shape

Fig. 17: Input point set, α-shape, Crust, nn-crust, χ-shape, Simple shape, Output of F. de Goes et al., ec-shape, wdm-crust, Crawl and result of our algorithm, along with its parameter(s). Each row corresponds to different features. The parts which are not detected well are marked in dotted circle. Our results are better or as good as the results of other algorithms.

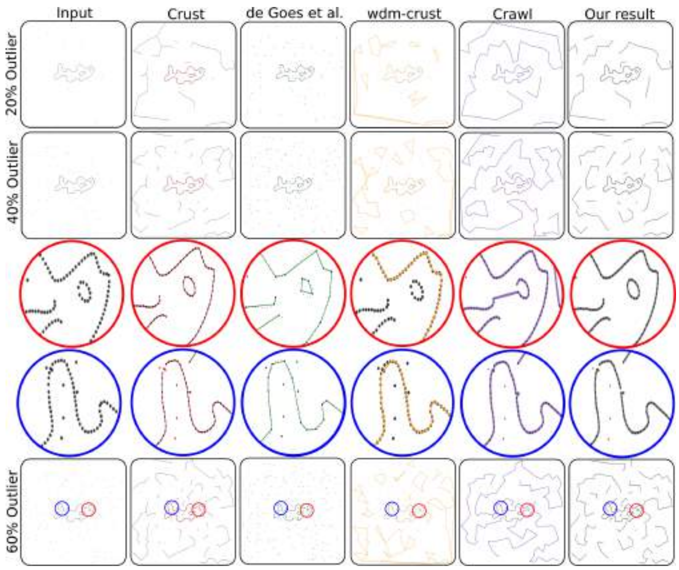


Fig. 18: Comparison with other algorithms- varying the outlier. 1^{st}, 2^{nd} and 5^{th} rows show the outputs with 20%, 40% and 60% outliers respectively. 3^{rd} and 4^{th} rows show the enlarged parts of the results in 5^{th} row. Our result is as good as those of crust and comparable or better than others even with 60% outlier.

is not able to capture the concave parts and disconnected components properly. Even though Crust, nn-crust and Crawl worked equally well in many of the cases, Crust and nn-crust have few unwanted edges in few cases whereas, Crawl could not capture many open curves. χ-shape and Simple shape capture the curve as such, however the corners and disconnected components are not captured well. Approach by F. de Goes et al. also gave good results in many cases but with less geometric details because of the presence of variable density. Empty circles stopped ec-shape from further digging and in many cases resulted in unvisited points. Wdm-crust is able to capture many details even though it could not make all points to be part of the curve.

Self-intersections: The last but one row of Figure 17 shows the results of various algorithms for an input with self-intersections. NN-crust and our algorithm reconstructs self-intersections better than the other algorithms.

Outliers: Similar to [20, 21, 22], our algorithm handles outliers. The last row of Figure 17 shows the result of various algorithms for an input with outliers. α-shape, nn-crust, χ-shape, Simple shape, ec-shape do not handle the outliers well. Extra edges in the curve are seen in the output of Crust. Even though the approach proposed by F. de Goes et al. [22] could obtain a comparatively good result, the reconstructed curve is an approximation with loss of geometry. Our result is as good as the results of wdm-crust and Crawl.

Figure 18 demonstrate the comparative study we performed varying the outliers. We have compared our results with those of Crust [10], deGoes et al. [22], wdm-crust [20] and Crawl [21]. First and second rows show the results for 20% and 40% outliers. Third and fourth rows show the enlarged parts of the circled portions of results for which the outliers is 60% (shown in the last row of Figure 18). Even with 60% outliers, our result performs as good as crust and comparable/ better than other methods.

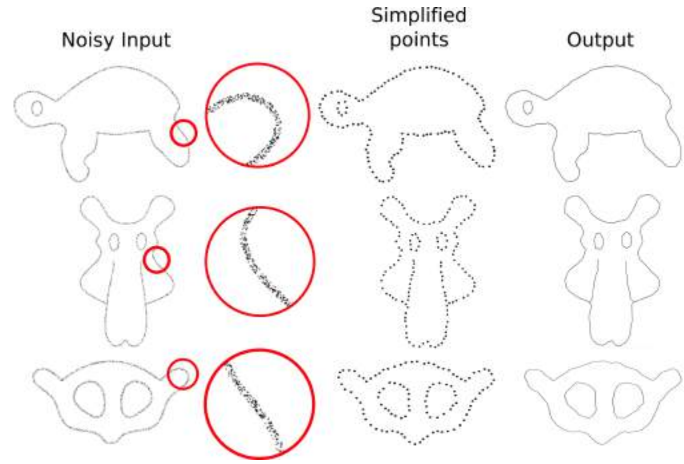


Fig. 19: Noisy point set (with noisy parts shown as enlarged), Point set after simplification (by our algorithm), Result of our algorithm. Self-intersections are also captured, as shown in second row.

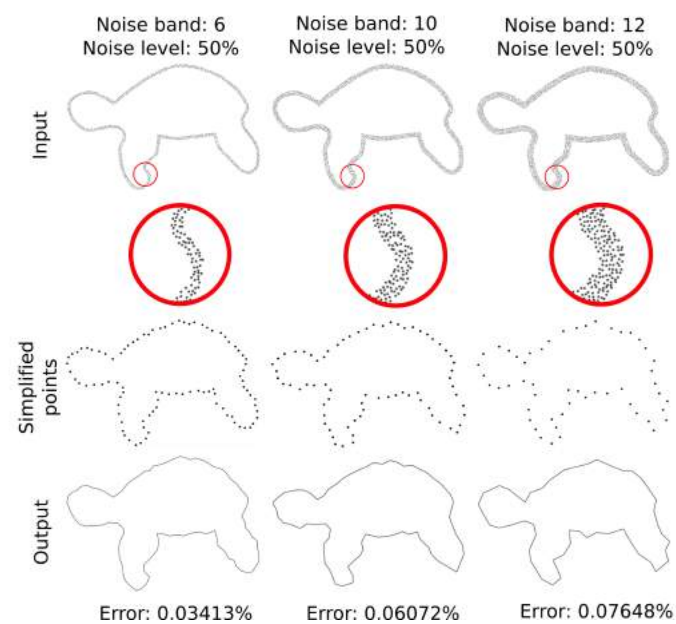


Fig. 20: Results of our algorithm varying the noise band with the error norm. The error increases when the value of noise band increases.

Noisy point set: We have taken the noisy inputs by sampling noisy line drawings. The sampling is done using WebPlotDigitizer tool, as in the case of a non-noisy point sets. Figure 19 shows noisy inputs (in the first column, with enlarged parts), point sets simplified by our algorithm (in the second column) and the results of our algorithm (in the third column).Our results reconstruct the curve well, even if the point set is noisy or the curves have self-intersections (second row of Figure 19). We also analysed the performance of our algorithm varying the noise level and noise band of the input point set.

Creating a noise band: For each point in the given line drawing of the input point set, a circle of radius r with centre as

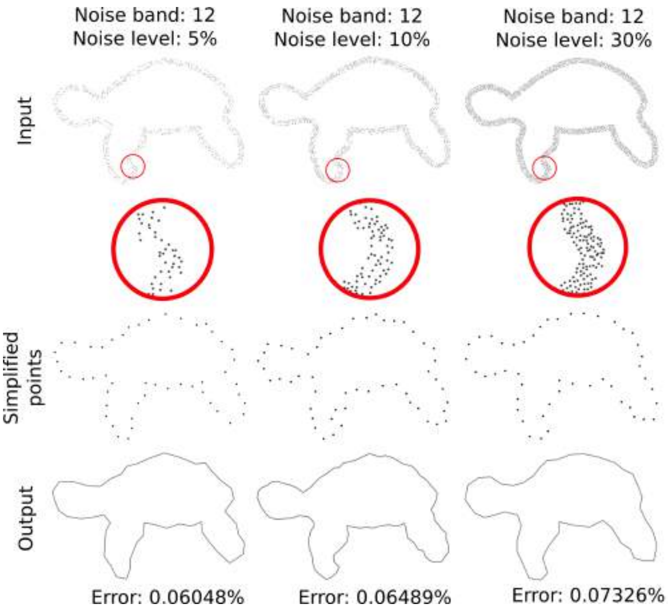


Fig. 21: Results of our algorithm varying the noise level with the error norm. The error increases when the value of noise level increases.

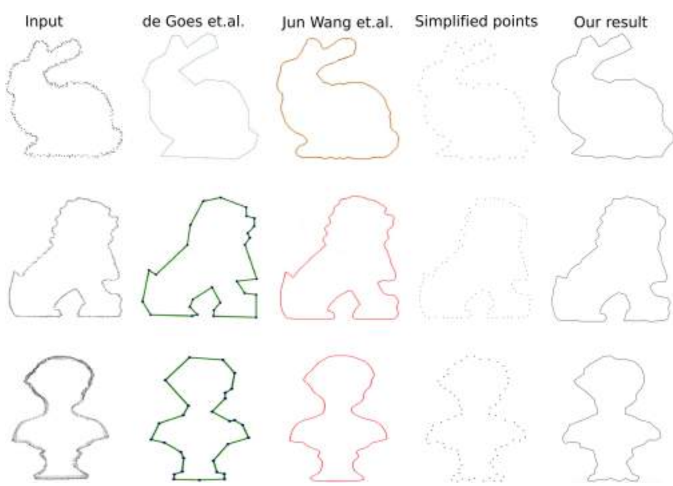


Fig. 22: For three objects: Noisy point set, Results of deGoes et al. [22] and Wang et al. [23], Point set after simplification (by our algorithm), Result of our algorithm

the point is considered. The union of all those circles of all the points is considered as a band area. A noise level $nl\%$ is induced in the band area thus creating a noise band.

Calculation of the error: Let R and O be the reconstructed and original simple closed curves respectively. Let $N(Q)$ be the number of points in the region Q, which is obtained by calculating the number of pixels by superimposing the images of the curves.

$$error = \frac{N((O - R) \cup (R - O))}{N(O)}$$

Figure 20 shows our results varying the noise band for a fixed noise level. Figure 21 shows our results varying the noise level, for a fixed noise band. The error increases when the value of noise band/noise level increases. Even though the error increases, the approximate shape of the curve is captured.

There are algorithms developed for handling noise [13, 14, 15], but they do not reconstruct open curves, disconnected components, curves with self intersections and are not designed for handling outliers. F. de Goes et al. [22] and Wang et al. [23] are the methods designed to handle noisy input, which reconstruct open curves and curves with self-intersections. Hence, for noisy input we have compared our results with only that of these methods.

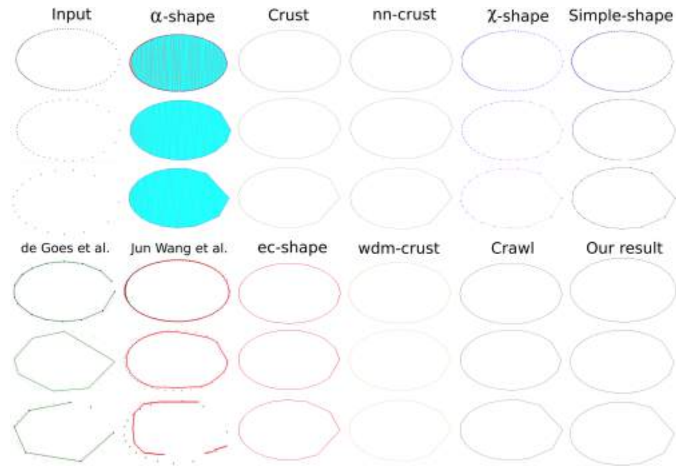


Fig. 23: Down sampled inputs (first three figures in the first column) and Results of other algorithms with ours. Results of F. de Goes et al. [22], Wang et al. [23] degrade its performance for down sampled inputs. Other algorithms perform as good as ours.

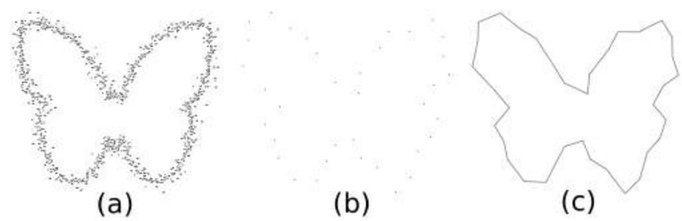


Fig. 24: (a) Scattered noisy point set (b) Point set after simplification (by our algorithm) (c) Our result where μ =110

Figure 22 shows sample noisy point sets along with outputs generated by F. de Goes et al. and Wang et al. The simplified point set generated by our noise simplification procedure and our reconstructed output are also shown in Figure 22. Our output capture geometry of the curve better than the result of F. de Goes et al. Algorithms introduced by F. de Goes et al. [22] and Wang et al. [23] are specially designed for noisy inputs. Figure 23 shows the output of various algorithms for simple closed curves with varying sampling density. Most of the algorithms performed reasonably well for the input that we tested except F. de Goes et al. and Wang et al. which reiterates the fact that they are tuned only for noisy inputs. Our algorithm captures the curve irrespective of the sampling density.

Please note that for parametric methods except F. de Goes et al. [22] and Wang et al. [23], we have tuned the parameter and selected the best results. For F. de Goes et al. [22] and Wang et al. [23], we have taken the results from their papers, along with the inputs.

Limitations: Even though our algorithm simplifies noisy inputs, if the noise is scattered, our algorithm yields only a coarse approximation of the curve, which is a limitation for Delaunay methods in general. Figure 24 shows the result of our algorithm for a scattered noisy point set. If the outliers are closer to the input points, our reconstruction produces extra lines as shown in the last columns of first, second and third and fifth rows of Figure 18. Also, we require one parameter each for simplifying the noisy point set and detection of self-intersections. Self-intersections are not identified automatically, only their restoration is performed by our algorithm.

4. Conclusion and Future work

In this paper, we proposed a simple curve reconstruction algorithm. Different from other works, we have designed a strategy to identify the presence of noise and hence, our algorithm is used for both simple and noisy point samples (with outliers too). The algorithm also restores self-intersecting curves. Our algorithm is a generalized one which is not input specific or feature specific. Experiments show that the algorithm works equally well or better than existing algorithms, detecting closed/open curves, disconnected components, sharp corners and multiple holes. Improving the output for scattered noisy point set and extending the work to higher dimensions are under consideration. Theoretical guarantee for our algorithm on noisy input and outputs with self-intersections are further future directions.

Acknowledgments

The authors would like to thank the reviewers for their valuable comments to improve the manuscript and Ministry of Human Resource & Development, Government of India for supporting the work.

References

[1] Edelsbrunner, H. Shape reconstruction with delaunay complex. In: Lucchesi, CL, Moura, AV, editors. LATIN; vol. 1380 of *Lecture Notes in Computer Science*. Springer. ISBN 3-540-64275-7; 1998, p. 119–132.

[2] Edelsbrunner, H, Kirkpatrick, DG, Seidel, R. On the shape of a set of points in the plane. IEEE Transactions on Information Theory 1983;29(4):551–558.

[3] Edelsbrunner, H, Mücke, EP. Three-dimensional alpha shapes. ACM Trans Graph 1994;13(1):43–72.

[4] Melkemi, M, Djebali, M. Computing the shape of a planar points set. Pattern Recognition 2000;33(9):1423–1436.

[5] Veltkamp, RC. Closed Object Boundaries from Scattered Points; vol. 885 of *Lecture Notes in Computer Science*. Springer; 1994. ISBN 3-540-58808-6.

[6] Duckham, M, Kulik, L, Worboys, MF, Galton, A. Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. Pattern Recognition 2008;41(10):3224–3236.

[7] Chevallier, N, Maillot, Y. Boundary of a non-uniform point cloud for reconstruction: extended abstract. In: Symposium on Computational Geometry. 2011, p. 510–518.

[8] Gheibi, A, Davoodi, M, Javad, A, Panahi, F, Aghdam, MM, Asgaripour, M, et al. Polygonal shape reconstruction in the plane. IET computer vision 2011;5(2):97–106.

[9] Attali, D. R-regular shape reconstruction from unorganized points. In: Proceedings of the Thirteenth Annual Symposium on Computational Geometry. SCG '97; New York, NY, USA: ACM. ISBN 0-89791-878-9; 1997, p. 248–253.

[10] Amenta, N, Bern, M, Eppstein, D. The crust and the beta-skeleton: Combinatorial curve reconstruction. In: Graphical Models and Image Processing. 1998, p. 125–135.

[11] Dey, TK, Kumar, P. A simple provable algorithm for curve reconstruction. In: SODA'99. 1999, p. 893–894.

[12] Amenta, N, Choi, S, Kolluri, RK. The power crust. In: Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications. SMA '01; New York, NY, USA: ACM. ISBN 1-58113-366-9; 2001, p. 249–266.

[13] Cheng, SW, Funke, S, Golin, M, Kumar, P, Poon, SH, Ramos, E. Curve reconstruction from noisy samples. Computational Geometry 2005;31(1):63 – 100. Special Issue on the 19th Annual Symposium on Computational Geometry - SoCG 2003.

[14] Mehra, R, Tripathi, P, Sheffer, A, Mitra, NJ. Visibility of noisy point cloud data. Computers and Graphics 2010;In Press, Accepted Manuscript:–.

[15] Feiszli, M, Jones, PW. Curve denoising by multiscale singularity detection and geometric shrinkage. Applied and Computational Harmonic Analysis 2011;31(3):392 – 409.

[16] Lee, IK. Curve reconstruction from unorganized points. Computer Aided Geometric Design 2000;17(2):161 – 177.

[17] Peethambaran, J, Muthuganapathy, R. A non-parametric approach to shape reconstruction from planar point sets through Delaunay filtering. Computer-Aided Design 2015;62:164 – 175.

[18] Methirumangalath, S, Parakkat, AD, Muthuganapathy, R. A unified approach towards reconstruction of a planar point set. Computers & Graphics 2015;51:90 – 97. International Conference Shape Modeling International.

[19] Methirumangalath, S, Kannan, SS, Parakkat, AD, Muthuganapathy, R. Hole detection in a planar point set: An empty disk approach. Computers & Graphics 2017;66:124–134.

[20] Peethambaran, J, Parakkat, AD, Muthuganapathy, R. A voronoi based labeling approach to curve reconstruction and medial axis approximation. 2015.

[21] Parakkat, AD, Muthuganapathy, R. Crawl through Neighbors: A Simple Curve Reconstruction Algorithm. Computer Graphics Forum 2016;.

[22] Goes, Fd, Cohen-Steiner, D, Alliez, P, Desbrun, M. An Optimal Transport Approach to Robust Reconstruction and Simplification of 2D Shapes. Computer Graphics Forum 2011;.

[23] Wang, J, Yu, Z, Zhang, W, Wei, M, Tan, C, Dai, N, et al. Robust reconstruction of 2d curves from scattered noisy point data. Comput Aided Des 2014;50:27–40.

[24] Favreau, JD, Lafarge, F, Bousseau, A. Fidelity vs. simplicity: a global approach to line drawing vectorization. ACM Transactions on Graphics (SIGGRAPH Conference Proceedings) 2016;.

[25] Berg, Md, Cheong, O, Kreveld, Mv, Overmars, M. Computational Geometry: Algorithms and Applications. 3rd ed. ed.; Santa Clara, CA, USA: Springer-Verlag TELOS; 2008. ISBN 3540779736, 9783540779735.

[26] Yvinec, M. 2D triangulation. In: CGAL User and Reference Manual; 4.10 ed. CGAL Editorial Board; 2017,.