CGI2018 manuscript No.
(will be inserted by the editor)

ConDT: A 2D curve reconstruction algorithm based on a constrained-neighbor proximity graph

J. Antony · M. Reghunath · S. B. Thayyil · R. Muthuganapathy

Abstract We introduce ConDT algorithm, a proximity-based reconstruction method relying on Delaunay Triangulation. The underlying proximity graph is referred to as the ConDT graph. In addition to being simple, the algorithm could successfully handle various challenging cases where classical reconstruction algorithms often struggle. Outlier removal is done in the post-processing phase using Interquartile Range (IQR) criteria, computed for the specific instance of the proximity graph. Relying on the recent benchmark on 2D reconstruction, we show that our method works better or is on par with the state-of-the-art methods.

Keywords Curve reconstruction · Delaunay triangulation · epsilon sampling · outlier handling

1 Introduction

Given a planar point set $S \in \mathbb{R}^2$ where $S = \{v_1, \dots, v_n\}$ sampled from an unknown curve Σ, the goal is to obtain "a piece-wise linear reconstruction of the curve", C from S that best approximates Σ. Curve reconstruction has received significant attention in computational geometry and computer graphics over the past two decades. Various algorithms [24] have been proposed to tackle this problem under different assumptions and conditions.

J. Antony · M. Reghunath · R. Muthuganapathy
Advanced Geometric Computing Lab, Department of Engineering Design,
Indian Institute of Technology Madras, Chennai, India
E-mail: jomsantony@gmail.com, minureghunath@gmail.com, emry01@gmail.com
Corresponding author: J. Antony

Safeer Babu Thayyil
Department of Information Technology, Government Engineering College Palakkad, India
E-mail: safeerbabut@gmail.com

Many of the early curve reconstruction algorithms assume that the input point set is sampled densely from a simple, smooth curve. Some well-known classical algorithms in this category include Crust, β-skeleton, NN-Crust, and α-shapes. In general, an input point set may exhibit characteristics such as non-uniform sampling, noise, and outliers. Meanwhile, the resulting curves can feature self-intersections, sharp corners, open ends, or disconnected components. Classical algorithms that provide theoretical guarantees assuming ϵ-sampling, may fail when the input point set contains the characteristic(s) mentioned above.

To handle these, newer algorithms have often resorted to tuning multiple parameters - a challenging task in general. Algorithms that do not use any parameter or use only a single parameter have difficulties in handling all the mentioned characteristics, as can be seen from Table 1 in [24]. In particular, handling multiple components, sharp corners, and outliers is challenging for such algorithms (refer to Table 1 in [24]).

Even the recent work [17] focuses only on manifold curves on a clean point set and is unable to handle outliers, open curves, curves with multiple components and non-manifold curves. Self-intersection is a common feature found in many planar curves. However, only a few methods [13,31,27] address its reconstruction.

This work, tested on the publicly available benchmark [24], with the following contributions:

- Introduces ConDT graph - a basic proximity graph structure which can be generated in a single-step parallel procedure from DT(S).
- Algorithm is demonstrated to handle self-intersections, multiple components, sharp corners (without any parameters), and open curves (with a single parameter).

– Outliers are handled using a dynamically generated IQR parameter specific to the ConDT proximity graph.

2 Related work

Edelsbrunner proposed a Delaunay triangulation-based parametric method to produce α-shape [11], which characterizes the shape of a point set. Although it was not originally designed for curve reconstruction, its 3D version [11] was later shown to be applicable for this purpose.

Crust algorithms [1,2] 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. [1], which is widely considered as a seminal work used to ensure theoretical guarantee of a reconstructed curve. Reconstruction using nearest neighbor graph with theoretical guarantee is presented in NN-Crust [7]. In Power Crust [2], 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. [5]. Mehra et al. [18] proposed a visibility operator on the convex hull of a noisy point set and used the visibility information to perform both curve and surface reconstructions. Feiszli et al. [12] introduced a non-parametric denoising strategy for reconstructing a curve while preserving sharp corners. However, the three curve reconstruction algorithms mentioned above do not reconstruct open curves, disconnected components, or curves with self-intersections, and they are not designed to handle outliers.

Lee [15] proposed a reconstruction method based on the moving least squares concept, specifically designed for noisy point sets to compute curves without self-intersections. Shape Hull [29] removes the edges of a Delaunay triangulation based on the position of the circumcenter of triangles to construct a simple closed divergent curve. Another Delaunay triangulation-based method, EC-Shape, uses the empty circle approach for outer boundary detection [20] and hole detection [19]. EC-Shape can reconstruct non-divergent curves but not open curves.

The Water-Distribution-Model (WDM) Crust [29] is based on the Voronoi diagram and handles outliers. Crawl [28] reconstructs closed/open curves with disconnected components and multiple holes; however, it does not handle noisy point sets. The Optimal Transport Cost method proposed by de Goes et al. [13] is a greedy method designed to minimize the increase in transport cost for noisy point sets. Wang et al. [31] proposed a quad-tree method with smoothing concepts to reconstruct a curve from a noisy point set with outliers. Most of the existing methods [10][29][20] are designed for simple closed curve reconstruction, whereas only a few of them [27][13][31] reconstruct both open and closed curves. A comprehensive survey and benchmarking of various algorithms for curve reconstruction are discussed by Ohrhallinger et al. [24]. This benchmark includes about 15 reconstruction algorithms, including the recent ones. Recently, deep learning approaches have also been explored for curve reconstruction, where networks are trained to predict and fit uniform B-splines to given point sets[4]

While several algorithms address aspects of curve reconstruction, it remains challenging to develop a generalized algorithm that handles all features: closed/open curves, disconnected components, self-intersections, multiple holes, and sharp corners, while also handling noise and outliers.

Although some reconstruction algorithms [20] [29][28] detect some of the mentioned features, they are not designed for handling noise. Algorithms specifically designed for handling noise [5] [18] [12] do not reconstruct open curves, disconnected components, or curves with self-intersections, and are not designed for handling outliers. Lee [15] designed a reconstruction algorithm for noisy point sets for both closed and open curves, but it is not capable of detecting self-intersections. F. de Goes et al. [13] and Wang et al. [31] claim to handle noise, self-intersections, and outliers, but their results on clean input point sets are very poor. Additionally, many algorithms have multiple parameters, making it tedious to synchronize and tune them.

3 Overview

Key concepts with regard to curve reconstruction such as medial axis, local feature size, and ϵ-sampling are well established in the literature and can be found in the seminal work by Amenta et al[1]. A quick review of these concepts can be found in Section 1 of the Supplementary. Now we introduce ConDT, the proximity graph employed in our method.

3.1 ConDT Proximity Graph

A ConDT graph (Constrained Natural Neighborhood Delaunay Triangulation) is a subgraph of a Delaunay Triangulation (DT) of a point set, S. For each point p, let $EN(p)$ denote the set of edges directly connected to

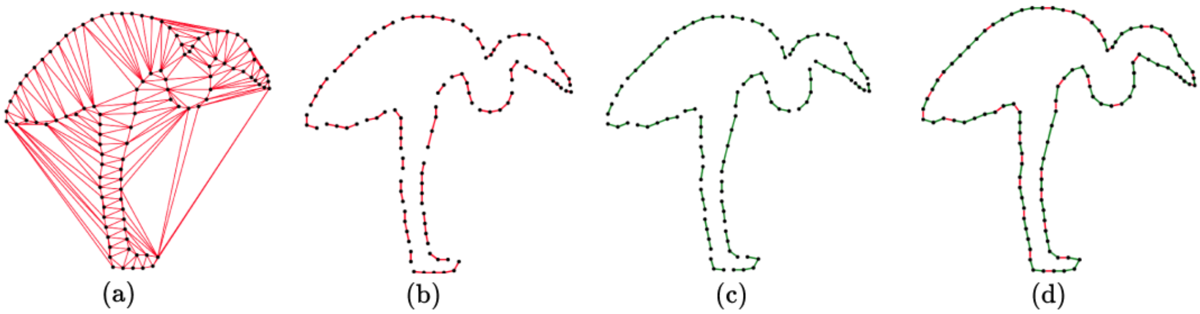


Fig. 1: Visualization of basic ConDT proximity graph construction (a) Input point set of bird and its DT (b) Shortest edge set say A from the natural neighbourhood of each point shown in red color (c) Second shortest edge set, say B in green color (d) Combining red and green edges to obtain our proximity graph ConDT which is $(A \cup B)$

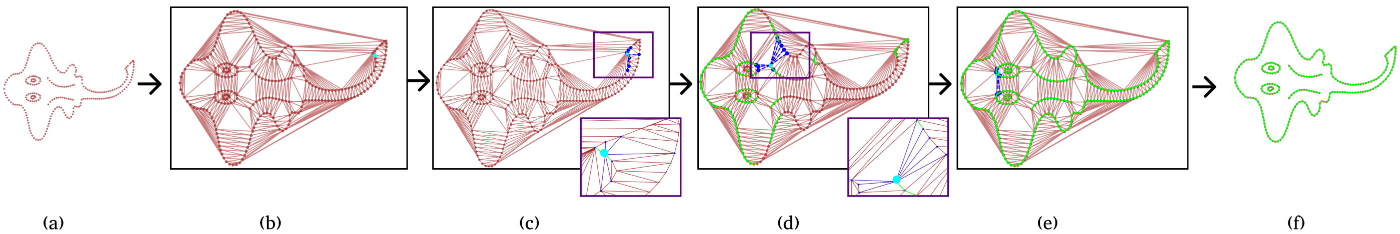


Fig. 2: Illustration of the proposed reconstruction algorithm on an input point set with multiple components and open curves in output. (a)Input point set (b) DT of the input point set (c) For an input point, p (in cyan), natural neighbours (shown in blue) with the selected edges (in green) (d) Illustration for a terminal point of an open curve (e) DT with reconstructed edges (in green) (f) Reconstructed output

p in the DT, referred as natural neighborhood edges. ConDT graph is formed by retaining only the shortest and second shortest edges from $EN(p)$ for each point p, resulting in a simplified and filtered graph formally defined as ConDT(S). This proximity graph produces a valid reconstruction when the input point set is well-sampled and free of outliers or noise.

Fig. 1 illustrates the construction of the ConDT graph. Fig. 1 (a) shows an input point set representing a bird shape adapted from the benchmark with additional points to ensure sufficient sampling. The process involves selecting the shortest edge shown in red in Fig. 1 (b) and the second shortest edge shown in green in Fig. 1 (c) from the natural neighborhood of each point. These selected edges are then combined into a set to form the resulting ConDT proximity graph, as depicted in Fig. 1 (d). This construction can be done in parallel across all the input points.

4 Method

The proposed method is outlined in the Algorithm 1. We start by constructing the DT of the input point set S. Identify the set of edges, $EN(p)$ in the natural neighborhood of each point p in the DT. From the set of edges in $EN(p)$, we retain only the first two shortest edges connected to each point to obtain ConDT(S). To take care of open curves (if any), one of the retained edges is to be removed based on a local uniformity parameter u. This parameter ensures the elimination of the longer edge by enforcing an allowable edge length ratio at each vertex.

The working of this algorithm is demonstrated with an example in Fig. 2. The input point set is shown in Fig. 2a along with its DT in Fig. 2b. For an input point p (in cyan), natural neighbourhood edges (in blue) and the selected edges (in green) are depicted in Fig. 2c. Illustration for another point is shown in Fig. 2d. in which only one green edge is selected, as the open curve criteria based on u parameter satisfies here. This basic reconstruction step is implemented in parallel for all points.

Instead of using a fixed value for u parameter, it can be adaptively computed on the go based on local neighbourhood as discussed in section 4.1

If reconstruction is limited to manifold curves, then we can handle non-manifold vertices in a post-processing step (Line 11). This procedure is illustrated in Algorithm 2 and discussed in section 4.2.

Outlier removal is done (Line 12) as a post-processing task and is illustrated in section 4.3.

Algorithm 1 ConDT($\mathbf{S}$)

Input: A planar point set $\mathbf{S} \subseteq \mathbb{R}^2$ representing the curve C, local uniformity parameter u.
Output: Reconstructed polygonal approximation, ∂O of the curve $\mathbf{C}$.

1: Compute the 2D Delaunay triangulation, $DT(\mathbf{S})$.
2: **for each** point $\mathbf{p}_i \in \mathbf{S}$ **in parallel do**
3: Collect the set of 1-ring vertices, $\mathcal{N}(\mathbf{p}_i)$, incident to $\mathbf{p}_i$ in $DT(\mathbf{S})$.
4: Collect the set of edges $EN(\mathbf{p}_i)$ connecting $\mathbf{p}_i$ to each of the vertices in $\mathcal{N}(\mathbf{p}_i)$.
5: Retain the two shortest edges, e_1 and e_2, in $EN(\mathbf{p}_i)$.
6: **if** $\max(e_1, e_2) \geq u \times \min(e_1, e_2)$ **then**
7: Retain only the shortest edge, $\min(e_1, e_2)$, in $EN(\mathbf{p}_i)$.
8: Combine the retained edges in $EN(\mathbf{p}_i)$ to form the edge set to obtain the proximity graph, $\mathbf{ConDT}(\mathbf{S})$.
9: HandleNonManifolds($\mathbf{ConDT}(\mathbf{S})$)
10: HandleOutliers($\mathbf{ConDT}(\mathbf{S})$)

Algorithm 2 HandleNonManifolds

1: **Input:** Edge set $\mathbf{ConDT}(\mathbf{S})$.
2: **Output:** Filtered edge set, *filtEdges*
3: **for** each edge $\{v_1, v_2\}$ in $\mathbf{ConDT}(\mathbf{S})$ **do**
4: **if** degree$(v_1) \leq 2$ **and** degree$(v_2) \leq 2$ **then**
5: Push edge $\{v_1, v_2\}$ to *filtEdges*
6: **for** each vertex v in $\mathbf{ConDT}(\mathbf{S})$ **do**
7: **if** degree$(v) \geq 3$ **then**
8: Push v to *NonManifoldVertices*
9: **for** each vertex v in *NonManifoldVertices* **do**
10: Find the shortest edge e_1 from v
11: Find the edge e_2 that gets the highest score based on angle and edge length criteria.
12: Push e_1 and e_2 to *filtEdges*
13: **return** *filtEdges*

4.1 Adaptive computation of uniformity parameter u

Selecting u based on local sampling density or edge length distributions allows the method to better accommodate variations in point spacing and curve structure. In this section, we outline a strategy that uses local neighbourhood information to compute the uniformity parameter dynamically.

This strategy adapts u based on the local distribution of edge lengths in the 1-ring neighborhood of each point, ensuring that edge comparisons remain proportional to the local scale. The adaptive computation is performed as follows.

For each point $\mathbf{p}_i$, we compute the set of edges in its 1-ring neighborhood, $EN(\mathbf{p}_i)$, and calculate the **mean** edge length:

$$\ell_i = \text{mean}\left(\{\|e\| : e \in EN(\mathbf{p}_i)\}\right)$$

The local threshold u_i is then computed as:

$$u_i = \frac{\ell_i}{\min(\|e_1\|, \|e_2\|)}$$

where e_1 and e_2 are the two shortest edges emanating from $\mathbf{p}_i$. The edges are retained if:

$$\max(\|e_1\|, \|e_2\|) < u_i \cdot \min(\|e_1\|, \|e_2\|)$$

This strategy tunes the the parameter u to adapt based on the local sampling density.

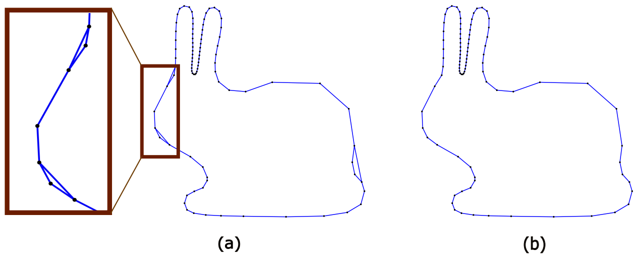


Fig. 3: (a) ConDT graph of a sparsely sampled input point set containing non-manifold vertices (some are shown in zoomed inset) (b) After removal of undesired edges using Algorithm 2

4.2 Handling Non-Manifold vertices

ConDT graph may end up with more than two edges from the same vertex as shown in Fig. 3 a. We refer to such vertices as non-manifold vertices. If we are reconstructing simple closed manifold curves, no vertex should have more than two edges attached to it. Non-manifold handling involves the identification of non-manifold vertices and retaining only the two most suitable edges. This is performed as a post-processing step and is detailed in Algorithm 2. Here we iterate through all non-manifold vertices and select at most two of them based on certain criteria.

We always choose the first shortest edge as it is guaranteed to be present in the result following the ϵ - sampling criteria in [7]. The second edge is chosen carefully based on a scoring mechanism. The edge selection score is given by

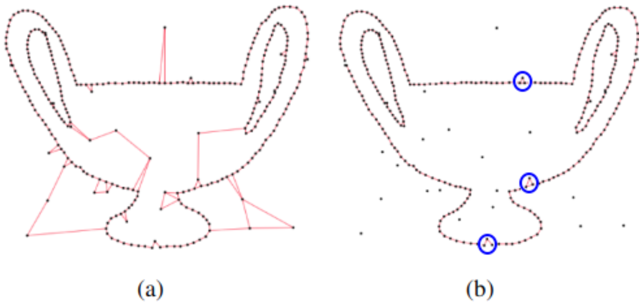


Fig. 4: (a) ConDT graph of input point set with outliers (b) Removal of outlier edges using IQR criteria

$$\text{score} = \max_{\substack{e \in E \\ \theta_e > \frac{2\pi}{3}}} \left(\frac{\theta_e}{\pi} \times \frac{L_{\min}}{L_e} \right)$$

where:

- $\frac{\theta_e}{\pi}$ is the normalized angle at the vertex where edge e is incident.
- L_e is the length of edge e.
- $L_{\min}$ is the shortest edge length among all candidate edges.

This score-based edge selection ensures the choice of an edge that forms a sufficiently large angle while favouring shorter edges for geometric consistency. The selection process enforces a minimum angle criterion of $\theta_e > \frac{2\pi}{3}$ to avoid choosing edges that are too sharp. We compute this score only for non-manifold vertices present in ConDT, unlike in NNCRUST [7] and HNNCRUST [21] where angle computation is done for every vertex. Fig. 3 illustrates how score-based selection helps in retaining only the suitable edges in the final output.

4.3 Handling Outliers

Algorithm 3 HandleOutliers

1: Let *sqlengths* be sorted list of squared edge lengths of **ConDT**(**S**).
2: $Q1 = \text{sqlengths}[n/4]$.
3: $Q3 = \text{sqlengths}[3n/4]$.
4: $\text{IQR} = Q3 - Q1$.
5: $IQR_{th} = Q3 + 1.7 \times \text{IQR}$.
6: **for each** vertex $\mathbf{v}_i \in$ **ConDT**(**S**) **do**
7:     **if** both outgoing edge lengths from $\mathbf{v}_i > 2.IQR_{th}$ **then**
8:         OUTLIERFOUND = true
9:         Break
10: **if** OUTLIERFOUND **then**
11:     Retain edges with squared lengths below the IQR_{th}.

Our proximity graph provides a good embedding of most of the retainable edges (part of the original curve) with some additional edges connected to the outlier points. Figure 4 (a) shows the ConDT graph of a point set containing outlier points.

The Interquartile Range (IQR) [14] method is used to filter out edges with unusually large lengths from the ConDT graph. The IQR threshold is computed from the sorted edge list. Edges with squared lengths below the computed IQR threshold are retained, effectively preserving edges that conform closely to the original shape, resulting in a better reconstruction as shown in Fig. 4b

This outlier removal is detailed in Algorithm 3. Testing various parameter values in the range $[1.0, 3.0]$, we observed optimal results within the interval $[1.7, 2.5]$, therefore we adopted 1.7 as the parameter value(refer to line 5 of Algorithm 3). Outlier points, generated using the benchmark implementation, include some placed very close to the curve, leading to a few undesired edges (see the regions enclosed in blue circle in Fig. 4b).

5 Results, Comparison & Discussion

We have tested and compared our method CONDT against 15 state-of-the-art curve reconstruction algorithms available in the benchmark implementation of "2D points curve reconstruction survey and benchmarking" [24] namely FITCONNECT [26], STRETCHDENOISE [25], CCRUST [8], PEEL [27], CRAWL [28], OPTIMALTRANSPORT [13], CONNECT2D [22], HNNCRUST [21], LENZ [16], CRUST [1], NNCRUST [7], GATHAN1 [9], GATHANG [6], DISCUR [32], VICUR [3] and also with the latest work SIGCONNECT [17]. For this we incorporated SIGCONNECT into the existing 2D Benchmark. OPTIMALTRANSPORT is omitted in some quantitative and qualitative evaluations, since it works only with input point sets with high level of noise and outliers and is unable to reconstruct clean point sets. We conducted a qualitative analysis by visually comparing the reconstructed curves, each exhibiting unique characteristics. For quantitative evaluation, we utilized metrics including exact reconstruction, RMS error, and runtime. The implementation is done using CGAL 5.6 [30] where parallel processing is enabled using Intel TBB (Threading Building Blocks) and tested on a system equipped with an Intel Core i7-12700 processor.

Table 1 shows the input and output capabilities of different algorithms along with their manifold guarantees, time complexity and capability of running on dense point set.

The results of ConDT on 3 different instances of simple closed curves, open curves, multiple curves, in-

Table 1: Comparison on input and output capabilities, time complexity, running times of various reconstruction algorithms. Under Input column notations used are NU: non uniform, NO: noisy, OU: outlier. The notations used under output column are O: open, MU: multiple components, S: sharp corners, SI: self intersections/non-manifold, G: guarantee. T: time complexity, Exactness: exact reconstruction

Algorithm	P	Input			Output					**T**	Dense Point Set	
		NU	NO	OU	O	MU	S	SI	G		Exactness	Run-time(ms)
FITCONNECT	0	yes	yes	yes	yes	yes	yes	no	yes	nk^2	yes	3503
STRETCHDENOISE	0	yes	yes	yes	yes	yes	yes	no	yes	nk^2	— Failed —	
CCRUST	0	yes	no	yes	yes	yes	no	no	yes	$nlogn$	yes	192
PEEL	2	yes	yes	yes	yes	yes	no	yes	yes	n^2	yes	749
CRAWL	0	yes	no	yes	yes	yes	no	no	no	$nlogn$	yes	34
OPTIMALTRANSPORT	0	yes	yes	yes	yes	yes	no	yes	yes	$nlogn$	no	222
CONNECT2D	0	yes	yes	no	no	no	yes	no	yes	$nlogn$	no	653
HNNCRUST	0	yes	no	no	yes	yes	no	no	yes	$nlogn$	yes	11
LENZ	2	yes	no	no	yes	no	yes	no	yes	$nlogn$	— Failed —	
CRUST	0	yes	no	no	no	yes	no	no	yes	$nlogn$	yes	10
NNCRUST	0	yes	no	no	yes	yes	no	no	yes	$nlogn$	no	4
GATHAN1	1	yes	no	no	yes	yes	yes	no	no	$nlogn$	yes	7
GATHANG	1	yes	no	no	yes	yes	yes	no	yes	$nlogn$	yes	78
DISCUR	0	yes	no	no	yes	yes	yes	no	yes	$nlogn$	— Failed —	
VICUR	4	yes	no	no	yes	yes	yes	no	no	$nlogn$	— Failed —	
SIGCONNECT	0	yes	no	no	no	no	yes	no	yes	$nlogn$	no	44
ConDT(Our's)	1	yes	no	yes	yes	yes	yes	yes	yes	$nlogn$	yes	6

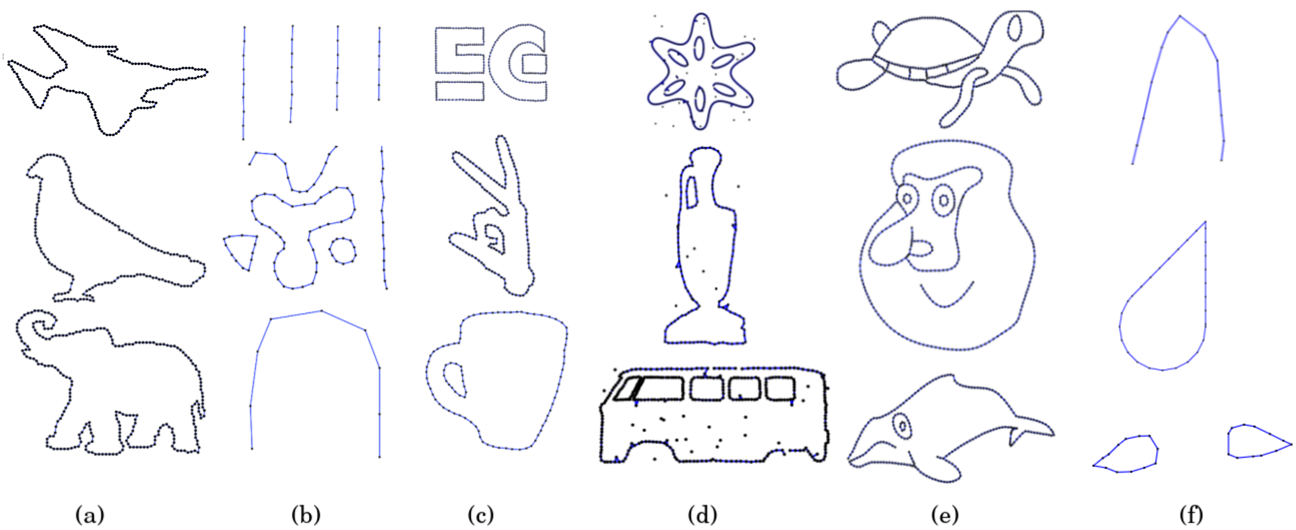


Fig. 5: Results obtained by ConDT on curves with different characteristics like (a) simple closed (b) open (c) multiple components (d) input with outliers (e) self intersections and (f) sharp corners.

put with outliers, curves with self-intersections, and curves with sharp corners are illustrated in Fig. 5.

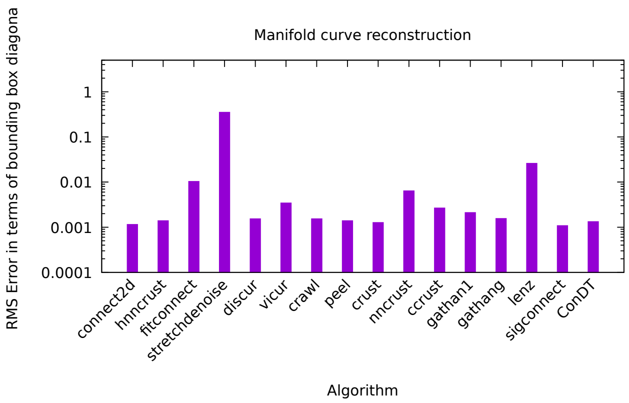


Fig. 6: RMS error of manifold curve reconstruction

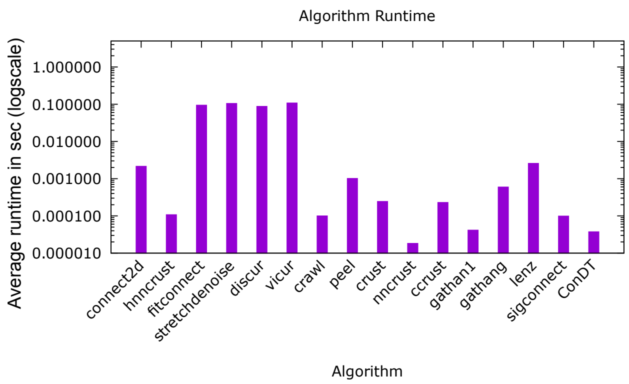


Fig. 7: Average runtime of manifold curves

Manifold curves: We selected a subset of 1,257 noise-free point sets representing manifold curves from the original benchmark dataset for comparison. This subset is chosen in such way that the ones with all input points are interpolated in the ground truth is only included. Plots comparing RMS error is shown in Fig.6. It can be noted that SIGCONNECT and CONNECT2D, which are optimized for manifold curve reconstruction, exhibited the minimum RMS error. However, they are not capable of reconstructing curves with other input and output features as indicated in table 1.

The plots in Fig. 6 show that ConDT is comparable to SIGCONNECT and CONNECT2D in terms of RMS error. The runtime plot in Fig. 7 illustrates that ConDT is on par with superior performers.

Well sampled Manifold curves: Sampling is applied on a few simple closed curves - bunny shape from benchmark and blob and simple shapes from [23] to

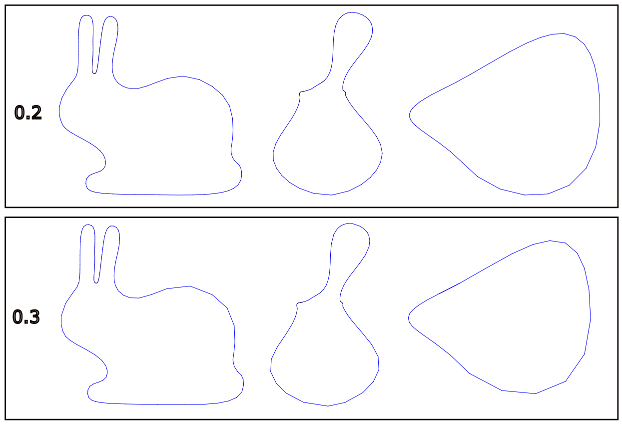


Fig. 8: Reconstruction of simple closed curves by ConDT with ϵ-sampling at ϵ=0.2 and 0.3

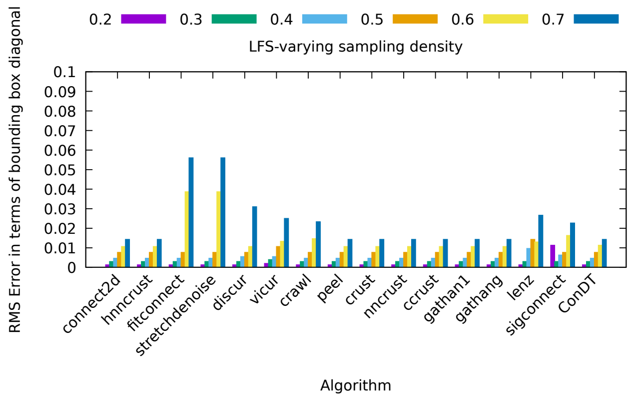


Fig. 9: RMS error of reconstruction for varying ϵ-sampling.

generate noise free point sets which follow the ϵ sampling criteria of $\epsilon = 0.2, 0.3, 0.4, 0.5, 0.6,$ and 0.7. Reconstruction of 3 simple closed curves by ConDT with ϵ-sampling at ϵ=0.2 and 0.3 is shown in Fig. 8. RMS error for reconstruction for varying ϵ sampling is shown in Fig. 9. ConDT is clearly superior against 7 algorithms and on par with other algorithms.

Densely sampled manifold curves:

Densely sampled point sets of over 10,000 points are obtained by sampling from the border samples available in the benchmark for testing.

The run-time and reconstruction exactness[24] of various algorithms are presented in Table 1. Despite the high sampling density, algorithms such as SIGCONNECT, CONNECT2D, NNCRUST, OPTIMALTRANSPORT, GATHAN and GATHANG failed to produce accurate reconstruction. Furthermore, algorithms like STRETCHDENOISE, LENZ, DISCUR, and VICUR were unable to generate the output. In contrast, our algorithm (a ba-

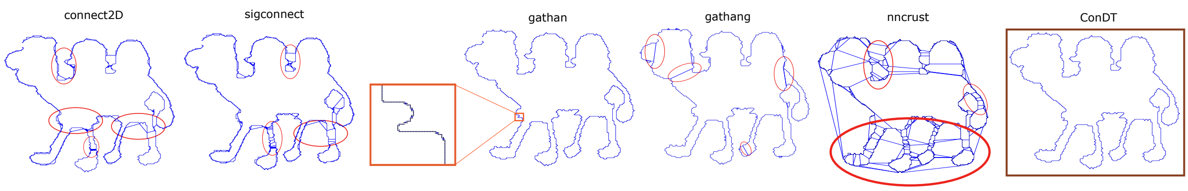


Fig. 10: Reconstruction on a densely sampled point set consisting of 10,518 points, our algorithm ConDT produced the exact reconstruction whilst algorithms like SIGCONNECT, GATHAN, GATHANG, NNCRUST generated incorrect results. (see the regions inside red ellipses) The Inset region for GATHAN shows break/incorrect connections.
.

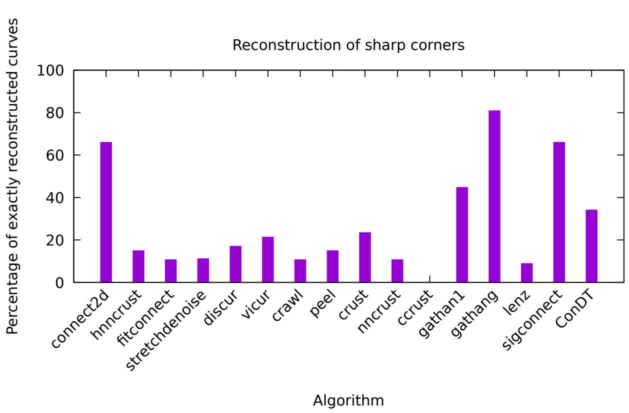


Fig. 11: Reconstruction of curves with sharp corners

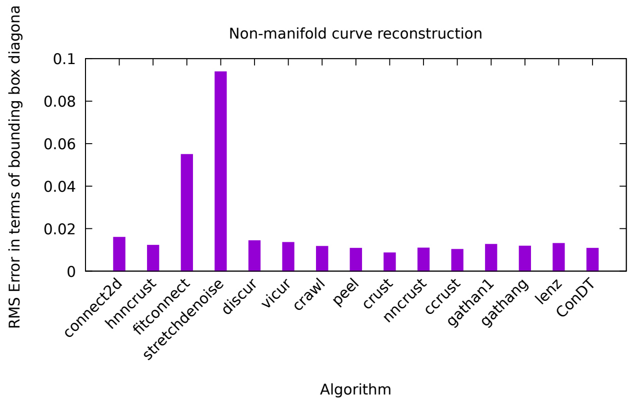


Fig. 12: Reconstruction of non-manifold/self-intersecting curves

sic version without non-manifold and outlier handling) successfully generated the correct output with a minimum runtime of 6ms. A qualitative comparison on a border sample representing the camel shape is shown in Fig. 10.

Sharp corners: We used 47 input point sets featuring sharp corners provided in the benchmark for comparison. The best results are obtained by GATHANG [6] and GATHAN [9] followed by CONNECT2D [22] and SIGCONNECT [17] (an improved version of CONNECT2D) which are specifically targeted at handling point sets with sparse sampling and sharp corners. Fig 11 presents the exactness plot, highlighting that while our algorithm is not specifically designed for sharp corners, it still achieved competitive performance, surpassing all the remaining algorithms.

Non-manifold/Self intersecting curves: Non-manifold /self-intersecting curves can be handled by our method (see Fig.1e). Fig. 12 presents the RMS error plot comparing different algorithms, demonstrating that ConDT performs competitively with the top-performing methods.

Open and multiple curves: Our method is capable of handling open and multiple curves with sufficient sampling (see Fig.1b and Fig.1c). The reconstruction for open curves can be fine-tuned by varying the local uniformity parameter u for each shape. Overall reconstruction exactness using a single parameter for all shapes is difficult to achieve. Here we used a common local uniformity parameter value, $u = 2.75$. However, we can evaluate the performance of different algorithms using a symmetric difference of area between the result obtained to the correct output[17]. This will help in validating that the reconstructed shapes closely resemble the expected output. The symmetric difference area (computed using BOOST's *boost_sym_diff*) between the output of different algorithms and the correct output is shown as a plot in Fig.13. It can be noted that the symmetric difference measure of ConDT is very low, second only to crust, which has the smallest value for open curves. RMS error comparison of different algorithms on multiple curves is shown in Fig. 14, it can be noted that the result is competitive with that of other algorithms.

We have also evaluated the performance of our method using an adaptively computed uniformity parameter, as described in Section 4.1. The comparative results using different algorithms with the adaptive uniformity parameter are provided in Section 2 of the Supplementary. The results are comparable to those obtained using a

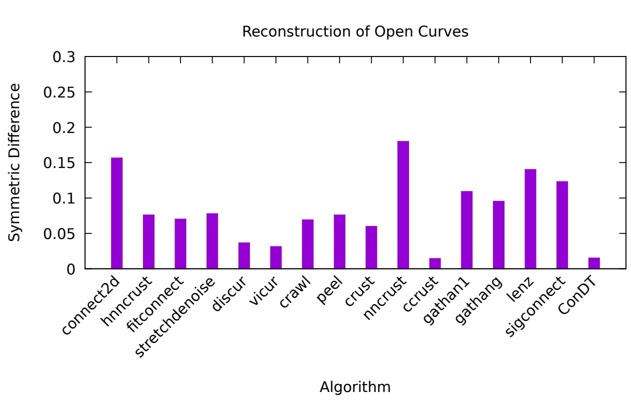


Fig. 13: Reconstruction of open-curves

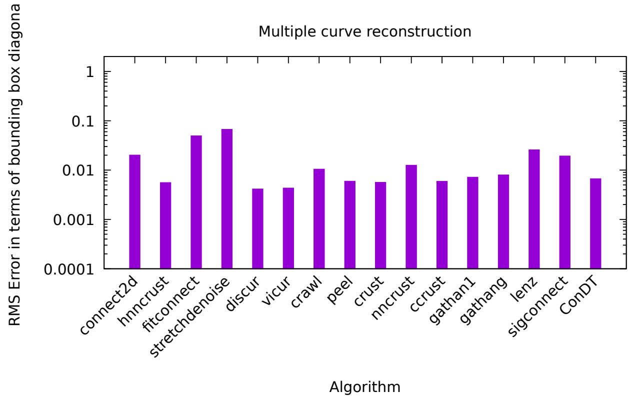


Fig. 14: RMS Error of reconstruction of multiple-curves

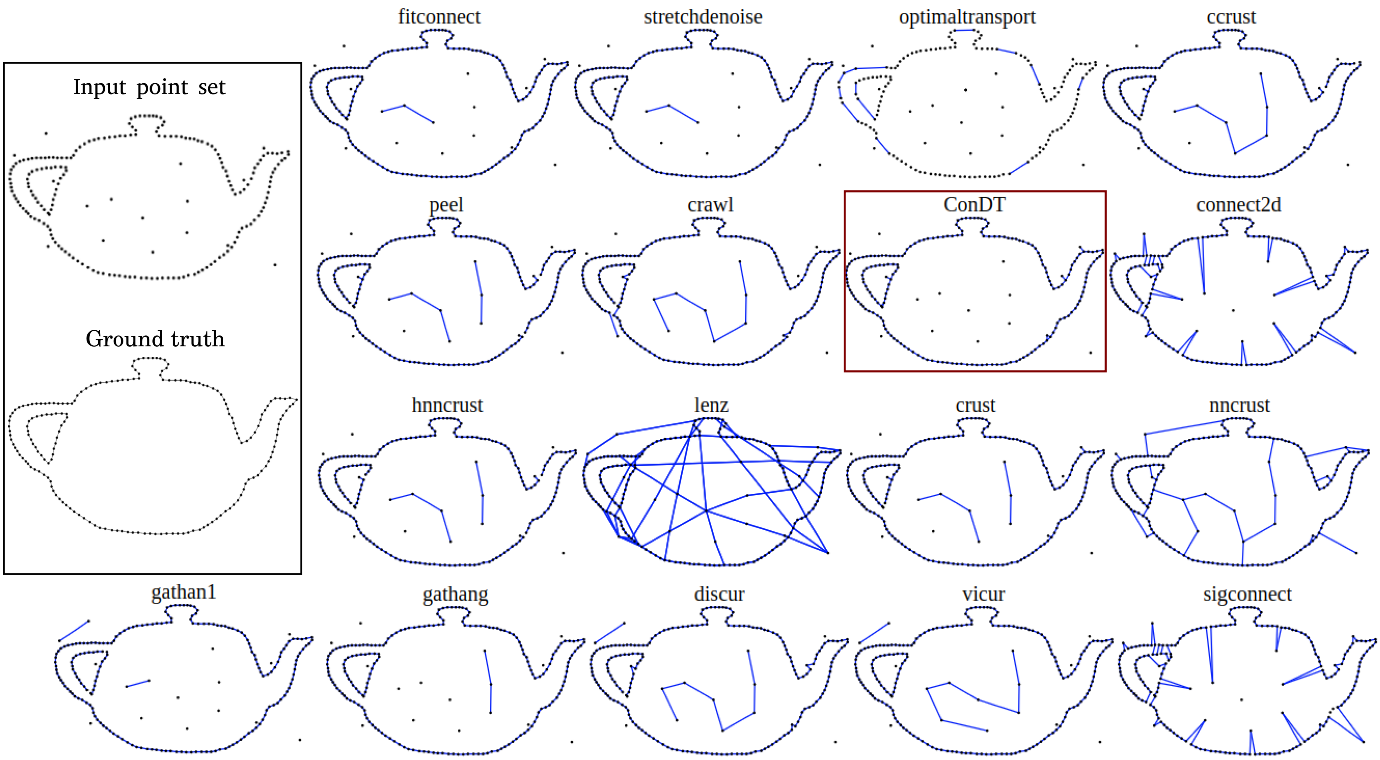


Fig. 15: Results of different curve reconstruction algorithms for input point set with 10% outliers. Input point set and ground truth is shown enclosed in black rectangle. ConDT result is enclosed in brown rectangle. Many long edges connected to outlier points are found in the output of other benchmark algorithms

carefully tuned u parameter. To fine tune the value u parameter, a sensitivity analysis is performed, RMS error is found to be minimum for u in range (2.5, 3.0). This plot is available in Section 2 of the Supplementary.

Outliers: Using the benchmark code[24], we evaluated our algorithm at different levels of outliers - 5%, 10% and 20% and observed that it outperforms all the existing methods. Fig. 15 demonstrates the superior performance of ConDT in reconstructing input point sets containing outliers. Notably, other reconstruction algorithms claiming to handle outliers either failed to remove many long edges (see results of FitConnect, StretchDenoise, CCrust, Peel, Crawl) or struggled to retain the desired edges (see OptimalTransport result). Algorithms like Connect2D, HNNCrust, Lenz, Crust, NNCrust, Gathan1, Gathang, Discur, Vicur and SigConnect do not have outlier handling capability, evident from the long undesirable edges in their reconstruction. Our algorithm has the minimum RMS error in all cases, as shown in Fig. 16. A value

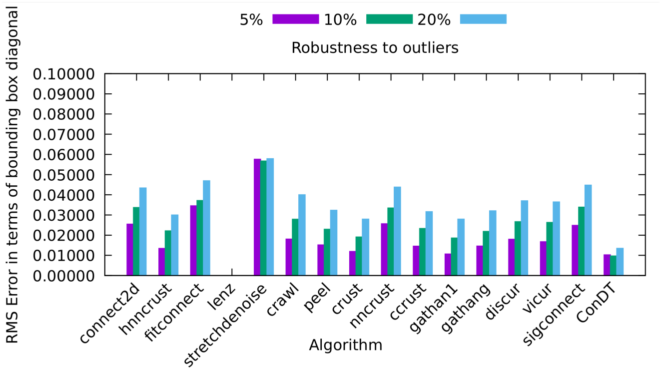


Fig. 16: RMS Error of reconstructed curves from the point set with 5%, 10% and 20% outliers added.

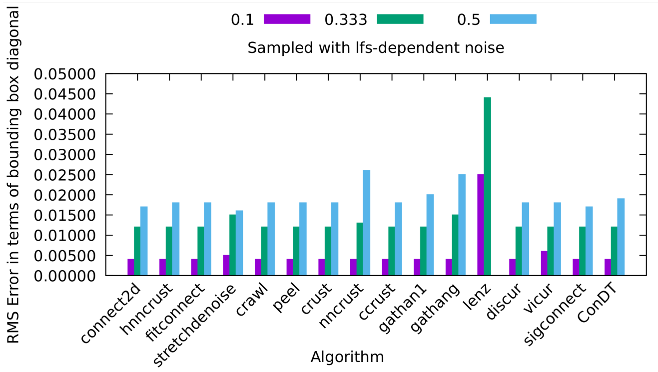


Fig. 19: RMS Error of reconstructed curves of points sampled with $\epsilon = 0.3$ and the points perturbed with local feature sized noise of $\delta = 0.1, 0.33$ and 0.5

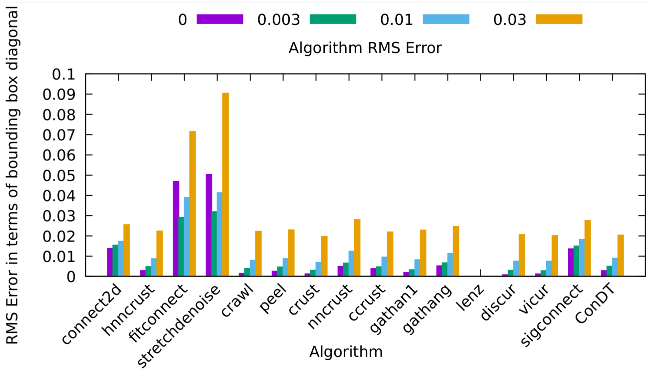


Fig. 17: RMS Error of reconstructed curves from the point set perturbed with uniform noise of $\delta = 0.003, 0.01$ and 0.03 as well as the non-noisy input.

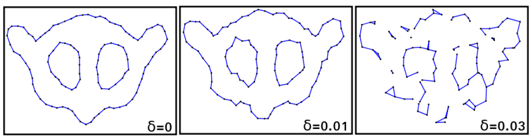
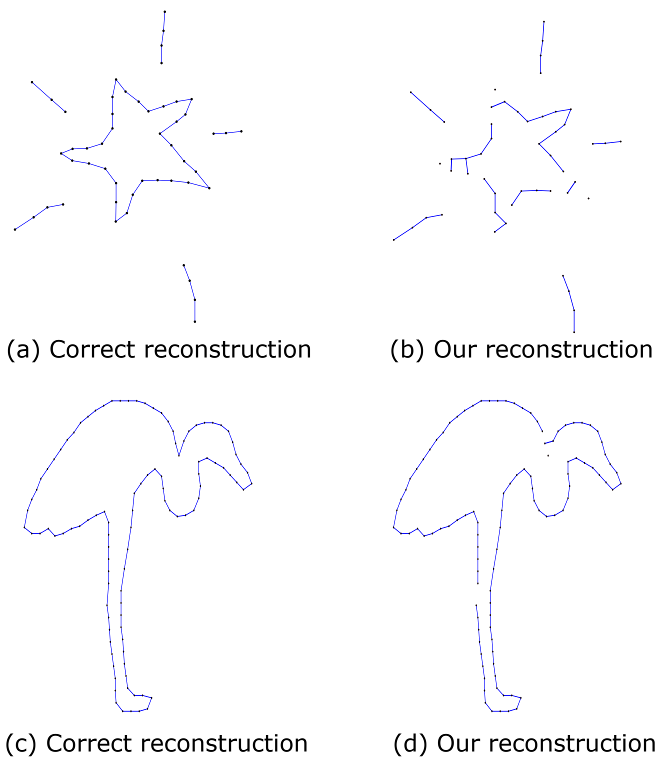


Fig. 18: Reconstructed curves with points perturbed with uniform noise of $\delta = 0$ (clean), 0.01 and 0.03. ConDT managed to reconstruct the original shape with $\delta = 0.01$ but failed to recreate the shape at $\delta = 0.03$ similar to all other state-of-the-art algorithms.



(a) Correct reconstruction (b) Our reconstruction

(c) Correct reconstruction (d) Our reconstruction

Fig. 20: A few failure cases

of 1.5 is generally chosen for IQR parameter as per statistics, but a sensitivity analysis for *IQR* parameter showed that a value of 1.7 is more appropriate. The sensitivity analysis plot for the IQR parameter is available in Section 2 of the Supplementary.

Noise: Robustness against noise is computed using the RMS error against the ground truth by introducing uniform noise levels of $\delta = 0.003, 0.01$ and 0.03 to the input curves. Here, noise level δ corresponds to the perturbation level with uniform noise as a percentage of the bounding box diagonal. The results are shown in Fig. 17. Our method ConDT exhibited the minimum RMS error in comparison with other state-of-the-art algorithms, even though our method is not explicitly designed for handling noise. Qualitative results for noise levels of 0 (noise-free), 0.01, and 0.03 for a curve with multiple components are illustrated in Fig. 18. Notably, ConDT managed to reconstruct the original shape with $\delta = 0.01$. For $\delta = 0.03$, our algorithm, as well as other state-of-the-art algorithms, failed to recreate the shape correctly. We also tested our method by

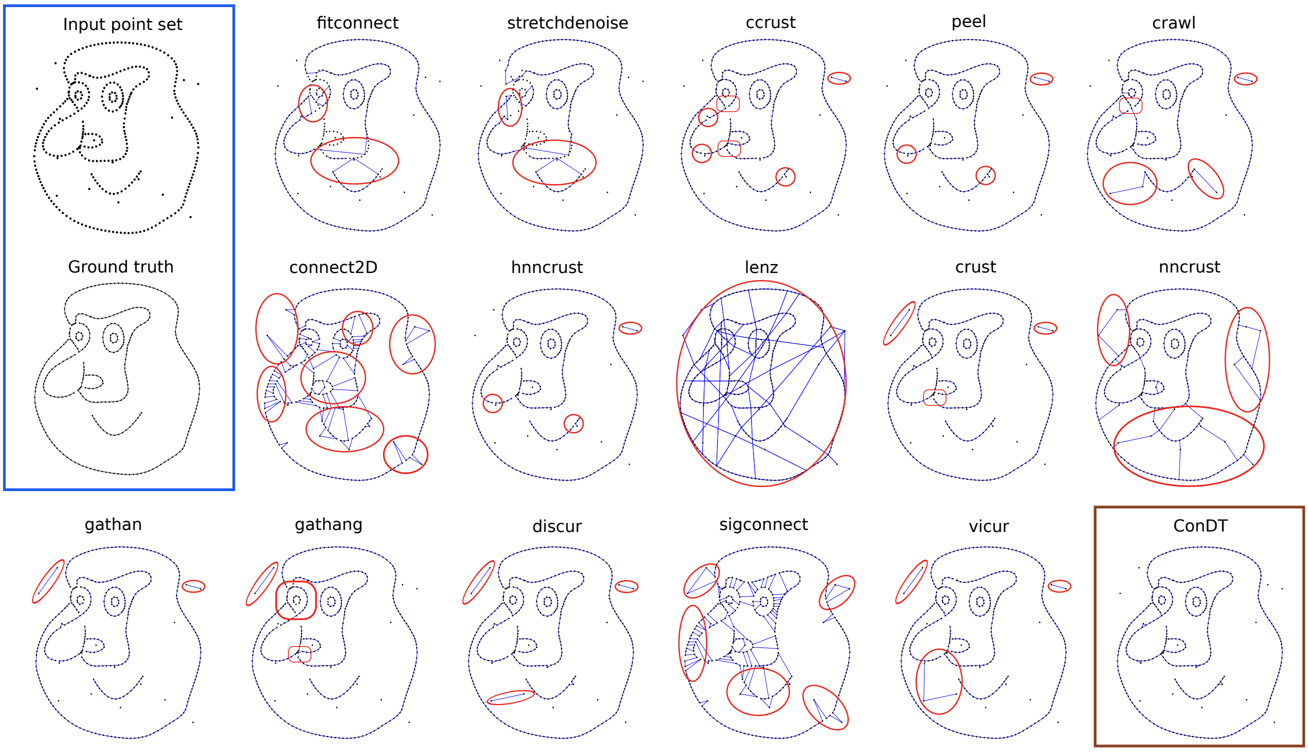


Fig. 21: Comparison of the results from different curve reconstruction algorithms applied to an input point set with various input characteristics, including (a) non-uniform sampling (b) presence of outliers (10%) and multiple output characteristics like (a) simple closed (b) open (c) multiple components or holes, (e) self-intersections. The input point set and ground truth are shown enclosed inside the blue rectangle. ConDT's output is shown enclosed inside a brown square. Regions enclosed in red ellipses indicate incorrect edges, red circles show connection breaks and red rounded rectangles highlight incorrect self-intersections (for other algorithms).

adding lfs noise on samples along a cubic Bézier curve keeping $\epsilon = 0.3$; results for the same are depicted in Fig. 19, which are better or competitive with other algorithms.

Summary: Unlike other state of the art algorithms, ConDT's uniqueness is that it is capable handling various input characteristics like (a) non-uniform sampling (b) presence of outliers, and multiple output characteristics like (a) simple closed (b) open (c) multiple components or holes, (e) self-intersections which is desirable in real use cases. The input point set depicted in Fig. 21. is an example that embeds all the input and output features listed. It can be noted that the reconstruction result by ConDT is superior in comparison with other state-of-the-art algorithms. The reconstructed outputs of other algorithms exhibit one or more of the following defects: (a) retained connections to outlier points enclosed inside red ellipses, (b) incorrect self-intersections enclosed in red circles, and (c) breaks in connections enclosed within red rounded rectangles.

6 Limitations

Our algorithm is not fine-tuned for handling noise, sharp corners, and sparse sampling. Although the current non-manifold and outlier handling approach works well in practice, it is not guaranteed to be error-free in all cases. Failure cases for outliers and noisy data are depicted in Fig. 4 (b) and Fig. 18 respectively. A few other failure cases are depicted in 20. Here (a) shows the correct reconstruction of a curve with multiple components and open segments, while (b) shows our reconstruction. (c) is the correct reconstruction of a simple closed curve, and (d) shows our result. It can be noted that our results are affected by sparse sampling, non-manifold processing and parameter tuning. Another failure case occurs when two independent curves are sampled in close proximity. This scenario is illustrated in Section 2, Figure 6 of the Supplementary material.

7 Conclusions and Future Work

We proposed a proximity graph-based reconstruction algorithm called ConDT. The ConDT proximity graph can be generated in a single parallelized step. The algorithm relies on a single parameter and is capable of reconstructing a wide range of characteristic curves. Our algorithm outperformed the state-of-the-art algorithms in outlier removal and noise handling in terms of RMS error. Moreover, it has exhibited superior performance in reconstructing highly dense point clouds in terms of exactness and runtime, as well as when multiple input and output features are present. In future work, ConDT can be further enhanced to explicitly address noisy input point sets and locally non-uniform sampling (e.g., regions with $\epsilon > 1$). Moreover, it can be extended to 3D for parallel surface reconstruction.

References

1. Amenta, N., Bern, M., Eppstein, D.: The crust and the beta-skeleton: Combinatorial curve reconstruction. Graphical Models and Image Processing pp. 125–135 (1998)
2. Amenta, N., Choi, S., Kolluri, R.K.: The power crust. In: Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications, SMA '01, pp. 249–266. ACM (2001)
3. An Nguyen, T., Zeng, Y.: Vicur: A human-vision-based algorithm for curve reconstruction. Robotics and Computer-Integrated Manufacturing **24**(6), 824–834 (2008). FAIM 2007
4. Barzegar Khalilsaraei, S., Komar, A., Zheng, J., et al.: Inceptcurves: Curve reconstruction using an inception network. The Visual Computer **40**, 4805–4815 (2024)
5. Cheng, S.W., Funke, S., Golin, M., Kumar, P., Poon, S.H., Ramos, E.: Curve reconstruction from noisy samples. Computational Geometry **31**(1), 63–100 (2005)
6. Dey, T., Wenger, R.: Fast reconstruction of curves with sharp corners. Int. J. Comput. Geometry Appl. **12**, 353–400 (2002)
7. Dey, T.K., Kumar, P.: A simple provable algorithm for curve reconstruction. In: SODA '99, pp. 893–894 (1999)
8. Dey, T.K., Mehlhorn, K., Ramos, E.A.: Curve reconstruction: Connecting dots with good reason. Computational Geometry **15**(4), 229–244 (2000)
9. Dey, T.K., Wenger, R.: Reconstructing curves with sharp corners. Computational Geometry **19**(2), 89–99 (2001). Combinatorial Curves and Surfaces
10. Duckham, M., Kulik, L., Worboys, M.F., Galton, A.: Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. Pattern Recognition **41**(10), 3224–3236 (2008)
11. Edelsbrunner, H., Mücke, E.P.: Three-dimensional alpha shapes. ACM Transactions on Graphics **13**(1), 43–72 (1994)
12. Feiszli, M., Jones, P.W.: Curve denoising by multiscale singularity detection and geometric shrinkage. Applied and Computational Harmonic Analysis **31**(3), 392–409 (2011)
13. de Goes, F., Cohen-Steiner, D., Alliez, P., Desbrun, M.: An optimal transport approach to robust reconstruction and simplification of 2d shapes. Computer Graphics Forum (2011)
14. Han, J., Kamber, M., Pei, J.: 2 - getting to know your data. In: J. Han, M. Kamber, J. Pei (eds.) Data Mining (Third Edition), The Morgan Kaufmann Series in Data Management Systems, third edition edn., pp. 39–82. Morgan Kaufmann, Boston (2012)
15. Lee, I.K.: Curve reconstruction from unorganized points. Computer-Aided Geometric Design **17**(2), 161–177 (2000)
16. Lenz, T.: How to sample and reconstruct curves with unusual features. In: Proceedings of the 22nd European Workshop on Computational Geometry (EWCG), pp. 3, 8, 12, 16, 20. Delphi, Greece (2006)
17. Marin, D., Ohrhallinger, S., Wimmer, M.: SIGDT: 2D curve reconstruction. Computer Graphics Forum **41**(7), 25–36 (2022)
18. Mehra, R., Tripathi, P., Sheffer, A., Mitra, N.J.: Visibility of noisy point cloud data. Computers and Graphics (2010). In Press, Accepted Manuscript
19. Methirumangalath, S., Kannan, S.S., Parakkat, A.D., Muthuganapathy, R.: Hole detection in a planar point set: An empty disk approach. Computers & Graphics **66**, 124–134 (2017)
20. Methirumangalath, S., Parakkat, A.D., Muthuganapathy, R.: A unified approach towards reconstruction of a planar point set. Computers & Graphics **51**, 90–97 (2015)
21. Ohrhallinger, S., Mitchell, S., Wimmer, M.: Curve reconstruction with many fewer samples. Computer Graphics Forum **35**(5), 167–176 (2016)
22. Ohrhallinger, S., Mudur, S.: An efficient algorithm for determining an aesthetic shape connecting unorganized 2d points. Computer Graphics Forum **32**(8), 72–88 (2013)
23. Ohrhallinger, S., Parakkat, A.D., Memari, P.: Feature-Sized Sampling for Vector Line Art. In: Pacific Graphics Short Papers and Posters. The Eurographics Association (2023)
24. Ohrhallinger, S., Peethambaran, J., Parakkat, A., Dey, T., Muthuganapathy, R.: 2D points curve reconstruction survey and benchmark. Computer Graphics Forum **40**, 611–632 (2021)
25. Ohrhallinger, S., Wimmer, M.: Stretchdenoise: parametric curve reconstruction with guarantees by separating connectivity from residual uncertainty of samples. In: Proceedings of the 26th Pacific Conference on Computer Graphics and Applications: Short Papers, PG '18, p. 1–4. Eurographics Association, Goslar, DEU (2018)
26. Ohrhallinger, S., Wimmer, M.: Fitconnect: Connecting noisy 2d samples by fitted neighbourhoods. Computer Graphics Forum **38**(1), 126–137 (2019)
27. Parakkat, A.D., Methirumangalath, S., Muthuganapathy, R.: Peeling the longest: A simple generalized curve reconstruction algorithm. Computers & Graphics **74**, 191–201 (2018)
28. Parakkat, A.D., Muthuganapathy, R.: Crawl through neighbors: A simple curve reconstruction algorithm. Computer Graphics Forum (2016)
29. Peethambaran, J., Muthuganapathy, R.: A non-parametric approach to shape reconstruction from planar point sets through delaunay filtering. Computer-Aided Design **62**, 164–175 (2015)
30. The CGAL Project: CGAL User and Reference Manual, 5.6 edn. CGAL Editorial Board (2023). URL https://doc.cgal.org/5.6/Manual/packages.html

31. Wang, J., Yu, Z., Zhang, W., Wei, M., Tan, C., Dai, N., Wei, L.: Robust reconstruction of 2d curves from scattered noisy point data. Computer-Aided Design **50** (2014)
32. Zeng, Y., Nguyen, T.A., Yan, B., Li, S.: A distance-based parameter free algorithm for curve reconstruction. Computer-Aided Design **40**(2), 210–222 (2008)