A unified approach towards reconstruction of a planar point set

Subhasree Methirumangalath, Amal Dev Parakkat, Ramanathan Muthuganapathy

Advanced Geometric Computing Lab, Department of Engineering Design, Indian Institute of Technology Madras
Chennai-600036, India.

Abstract

Reconstruction problem in $\mathbb{R}^2$ computes a polygon which best approximates the geometric shape induced by a given point set, S. In $\mathbb{R}^2$, the input point set can either be a boundary sample or a dot pattern. We present a Delaunay-based, unified method for reconstruction irrespective of the type of the input point set. From the Delaunay Triangulation (DT) of S, exterior edges are successively removed subject to *circle* and *regularity* constraints to compute a resultant boundary which is termed as *ec-shape* and has been shown to be homeomorphic to a simple closed curve. Theoretical guarantee of the reconstruction has been provided using r-sampling. In practise, our algorithm has been shown to perform well independent of sampling models and this has been illustrated through an extensive comparative study with existing methods for inputs having varying point densities and distributions. The time and space complexities of the algorithm have been shown to be $O(n\,log\,n)$ and $O(n)$ respectively, where n is the number of points in S.

Keywords: Delaunay Triangulation, Reconstruction, Dot Pattern, Boundary Sample.

1. Introduction

Given a finite set of points $S \subseteq \mathbb{R}^2$, reconstruction problem computes a polygon which best approximates the geometric shape induced by S [1]. The major challenges of the reconstruction problem are the facts that it is ill-posed and there is little success in phrasing it as an optimization problem [1]. It is an extensively studied problem because of the existence of varied applications and the application specific nature of the output [2]. Quantifying how much the output approximates S is a difficult task [1] and thus there are different outputs for the same point set. The output highly differs with human cognition and perception [1] and it is dependent on heterogeneity in density and distribution of S.

Algorithms for reconstruction are based on the sampling of the input shape, which is of two types. One category of input consists of points sampled only from the boundary of the object, termed as boundary sample [3] or curve sample [4], as shown in Figure 1(a). The other category consists of points sampled from the whole object termed as dot pattern [3] or object sample [4] as shown in Figure 1(b). We use RBS to denote reconstruction from a boundary sample (Figure 1(c)) and RDP for reconstruction from a dot pattern (Figure 1(d)) respectively.

Algorithms for reconstruction can also be classified as two types: Delaunay based and non-Delaunay methods. As our algorithm is Delaunay based, we focus our discussion mainly on Delaunay based methods. One of the earliest attempts to characterise a set of points in the plane was by Edelsbrunner et al.'s α-shape [5]. Another one (though in 3D) is the sculpting algorithm by Boissonnat [6]. In [7], sculpting strategy is based on the length of the boundary edge of tetrahedron where as in [4], it is based on the circumcircle of an exterior triangle. The reconstruction in [8] is done by a greedy simplification of Delaunay triangulation using a series of half-edge collapse operations that minimizes the increase of total transport between the input point set and the triangulation. Galton and Duckham proposed an algorithm for characteristic shape (χ-shape), where the longest edge from the Delaunay triangulation was removed if it satisfied certain conditions [9]. Family of crust algorithms based on Delaunay Triangulation were introduced to capture various features of a point set [10, 11, 12]. Regular interpolants, which are the polygonal approximations of planar curves are introduced in [13]. RBS based on γ-graph is presented in [14]. Ball pivoting algorithm [15] is a non-Delaunay method which starts with a seed triangle and proceeds by pivoting a ball to get the next point. Join and glue are the main topological operations performed in the algorithm which adds and deletes edges respectively. Simple shape algorithm [3] presents a non-Delaunay

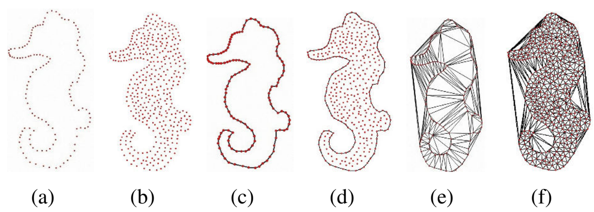


Figure 1: (a) Boundary sample. (b) Dot pattern. (c,d) Reconstructed shapes (*ec-shapes* in this paper) of the boundary sample and the dot pattern. (e) Delaunay triangulation of boundary sample. (f) Delaunay triangulation of dot pattern.

Email addresses: `subhasree.rajiv@gmail.com` (Subhasree Methirumangalath), `adp.upasana@gmail.com` (Amal Dev Parakkat), `emry01@gmail.com` (Ramanathan Muthuganapathy)

approach for reconstruction that can handle both dot patterns as well as boundary samples.

1.1. *Motivation*

In general, reconstruction, irrespective of the type of input, has many applications in various fields. Reconstructed boundary unambiguously defines a valid object on these points and can be used for initial design of an artifact, for numerical analysis, or for graphical display [14]. Map generalization [2] is one among many other applications of reconstruction in the field of Geographical Information Systems (GIS).

It is to be emphasised that RBS has an equivalent problem in three-dimension (3D) which is popularly known as surface reconstruction where as the problem of reconstruction from dot pattern has no equivalent problem in 3D. Hence, the reconstruction problem is very much relevant in two-dimensions (2D) itself as the host of recent applications (such as GIS and biomedical image analysis) indicate. Almost all the approaches for reconstructing from either type of input depend on at least one input parameter which is difficult to identify. Moreover, most of the current approaches deal with only one of the input type and not both (except [3]). The approaches that work for RBS may not work for RDP and vice versa, illustrating the requirement of a unified approach. Most algorithms have been tuned to work only for one kind of input (such as Crust [10], which has been tuned for boundary samples). Hence, the major motivation in this paper is to provide an approach for reconstruction that is independent of the nature of the input.

Our algorithm differs mainly from other sculpting algorithms in its sculpting strategy. We use circle (three types of circles) and regularity constraints as the strategy where as in [4] it is based on a combination of circumcenter and circumradius of Delaunay triangle. An optimal transport-driven approach is proposed in [8]. The constraints imposed on removing an edge depends on Euclidean Minimum Spanning Tree and Extended Gabriel Hypergraph in [7]. In [15], decision on whether to insert an edge to the boundary is made using a pivoting ball where as simple shape algorithm in [3] replaces a selected edge of initial convex hull with two new edges using a selection criteria value which depends upon edge length, closeness of points and angle formed by the two new edges. In [6] the sculpting strategy is based on the maximum distance in the sculpture.

1.2. *Our contributions*

- A unified approach for RBS as well as RDP has been proposed.
- An empty circle approach using DT has been proposed.
- Theoretical guarantee as well as extensive experiments have been provided to evaluate the proposed approach.
- Demonstrated that the approach works well where other algorithms have restrictions.

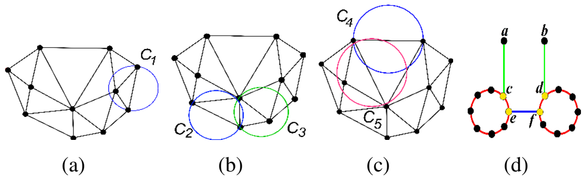


Figure 2: (a) Non-empty diametric circle C_1 (b)Empty diametric circle C_2 and non-empty chord circle C_3 (c)Empty diametric circle C_4 and non-empty midpoint circle C_5 (d) Regularity constraint

2. Preliminaries

Let $S = \{p_1, p_2, p_3, ..., p_n\} \subseteq \mathbb{R}^2$ be the input point set of n points. The line segment between two points p_i and p_j, including its end points, is termed as an *edge*, denoted as e_{ij}. $\triangle_{ijk}$ denotes the triangle formed by three points p_i, p_j and p_k. Delaunay Triangulation of the input point set S (which is a hyper graph of S) is denoted as $DT(S)$. DTs of boundary sample and dot pattern are shown in Figures 1(e) and 1(f) respectively.

DEFINITION 1 Exterior Triangle (ET) of a graph is a triangle which has at least one edge which is not shared by any of the other triangle.

DEFINITION 2 Exterior edge (EE) of an ET is the edge not shared by any other triangle in the graph. A vertex of an exterior edge is called an exterior vertex.

DEFINITION 3 Chord circle of an edge e_{ij} is a circle with e_{ij} as its chord.

DEFINITION 4 Midpoint circle of an edge e_{ij} is any circle whose centre is the mid point of the edge.

DEFINITION 5 Diametric circle of an edge e_{ij} is a midpoint circle with diameter $\|p_j - p_i\|$.

Figure 2(a) shows a diametric circle C_1 for an EE of an ET. It is to be noted that a diametric circle is associated only with an EE (of an ET), where as the chord and midpoint circles are always associated with adjacent sides of the EE of the ET. A chord circle C_3 having the same radius of diametric circle C_2 is shown in Figure 2(b). Two chord circles are possible using the radius of C_2 on the same edge. A midpoint circle C_5 having the same radius of diametric circle C_4 is shown in Figure 2(c).

3. Algorithm

3.1. *Algorithm Idea*

Consider a diametric circle of an exterior edge e (Figure2(a)). The intuition is that, if the diametric circle is non-empty, then e is comparatively longer in the local neighbourhood. Even if the diametric circle of e is empty and the chord circle(s) or midpoint circle(s) of the adjacent sides of ET (whose exterior edge is e) is non-empty, then e is comparatively longer in the local neighbourhood. The non-emptiness of any of the three types of circle(s) indicates that the vertices of e might not be neighbours in the boundary of the original shape and e can be removed from the graph.

3.2. Regularity and Circle Constraints

DEFINITION 6 A dangling edge e in G is a bridge [16] such that G-e has exactly one more component than G and one of the components in G-e is an isolated vertex, where G-e denotes G without e.

Figure 2(d) illustrates a graph containing dangling edges (e_{ac} and e_{bd}), bridge (e_{ef}) as well as junctions points (c, d, e and f). Junction point is also known as cut vertex [16]. It is obvious that all dangling edges are bridges.

Regularity Constraint - A graph is said to be regular if it does not have bridges, dangling edges or junction points.

Circle Constraint - The exterior edge of an ET in a graph is said to satisfy circle constraint if any one of the following conditions is satisfied:

- Diametric circle (say, radius R) of the exterior edge of the graph is non-empty (i.e., the circle contains at least one point of S).
- Any chord circle with the same radius R for any of the adjacent sides of the ET is non-empty (a chord circle is available when $2R >$ *the length of the adjacent side*).
- Any midpoint circle with the same radius R for any of the adjacent sides of the ET is non-empty (a midpoint circle is available when chord circles are not available ie. when $2R <=$ *length of the adjacent side*).

3.3. Algorithm details

The algorithm consists of two steps; (a) Removing an exterior edge (EE) (and hence the ET) and (b) Check for termination.

Removing an exterior edge

Initially, the graph (say, G) is $DT(S)$. The exterior edges of G are arranged in a priority queue (PQ) in the descending order of the edge lengths. First EE is taken from the PQ and checked for *circle constraint*. If it satisfies the constraint, then the graph $G - EE$ (ie. G with out the EE) is checked for regularity. G is then updated to $G - EE$, if $G - EE$ is regular. Broadly the exterior edge is removed if it satisfies the circle constraint and the graph without the edge is still regular. Removing an exterior edge implies that the corresponding ET is also deleted from the graph. The adjacent edges (which are still edges in some other triangles in the graph) of the removed ET are updated to exterior edges and added to the PQ, maintaining the descending order of the edge lengths.

Check for termination

An EE cannot be removed if it does not satisfy the circle constraint or the graph excluding the EE is not regular. The algorithm terminates when there is no possibility of removing any EE.

Algorithm 1 gives the pseudocode for generating ec-shape, given a point set S. Time complexity of our algorithm depends on construction of DT, construction of PQ and its updation, checking circle and regularity constraints and removal of an EE from G. The non-emptiness of any circle of an EE implies presence of at least one vertex of the adjacent triangles of its ET. Hence checking circle constraint for EE of ET takes constant time because it is enough to check the points of two adjacent triangles of the ET. An ET is considered for removal only if it has exactly one vertex which is not exterior, which can be easily done by setting a flag for exterior vertices. Hence, regularity constraint can be ensured by checking the flag of the third vertex of ET(the vertex not part of EE) and it can be done in constant time. Removal of EE from G is of constant time because it is done by ensuring circle and regularity constraints. The two edges which replaces EE becomes part of PQ and one updation of PQ takes $O(log\, n)$ time. The number of edges of DT is $O(n)$ and hence the overall updation of PQ takes $O(n\, log\, n)$ time. Initially, DT and PQ are constructed in $O(n\, log\, n)$ time and hence over all complexity of our algorithm is $O(n\, log\, n)$. As no extra space is needed for performing any of the steps in the algorithm, its space complexity is $O(n)$.

Algorithm 1: ec-shapeConstruction(S)

Input: Input point set, S.
Output: ec-shape.

Construct a graph $\mathfrak{G}$ = Delaunay Triangulation, $DT(S)$.
Construct a Priority Queue (PQ) of EEs of $\mathfrak{G}$ in the descending order of edge lengths.
repeat
    Delete the EE of ET from the head of PQ and remove it from $\mathfrak{G}$, if it satisfies the *circle* constraint and $\mathfrak{G} - EE$ is regular.
    If EE is removed from $\mathfrak{G}$, add the adjacent sides of the ET to PQ maintaining the descending order of the edge lengths.
until No more EE in $\mathfrak{G}$ can be removed.
return ec-shape, the exterior edges of the graph $\mathfrak{G}$.

3.4. Illustration of Algorithm

Figure 3 illustrates Algorithm 1 using the dot pattern shown in Figure 3(a). The DT, which is the initial graph G for the dot pattern, is shown in Figure 3(b). In this section, we denote an edge as e_i, for convenience. The exterior edges are then put in priority queue in the descending order of their lengths. The longest one is picked at the beginning (e_1 in Figure 3(c)). The diametric circle of e_1 satisfies circle constraint and $G - e_1$ is regular. e_1 is then removed (and its corresponding ET) and edges e_2 and e_3 are updated (Figure 3(d)) as EEs and are added appropriately in the queue. G is updated to $G - e_1$. Algorithm proceeds further and removes few more EEs (Figure 3(e)) along with updating G. When the algorithm encounters the edge e_4, the diametric circle of e_4 (Figure 3(f)) is empty. Since e_4 is shorter than the other two edges of the ET, no chord circles are available. Hence, the midpoint circles (blue color in Figure 3(f)) are used for testing the circle constraint using the radius from the diametric circle of e_4. As one of the midpoint circles satisfies circle constraint, and $G - e_4$ is regular, e_4 is removed. The edges e_5 and e_6 (Figure 3(g)) are updated to EEs (and their

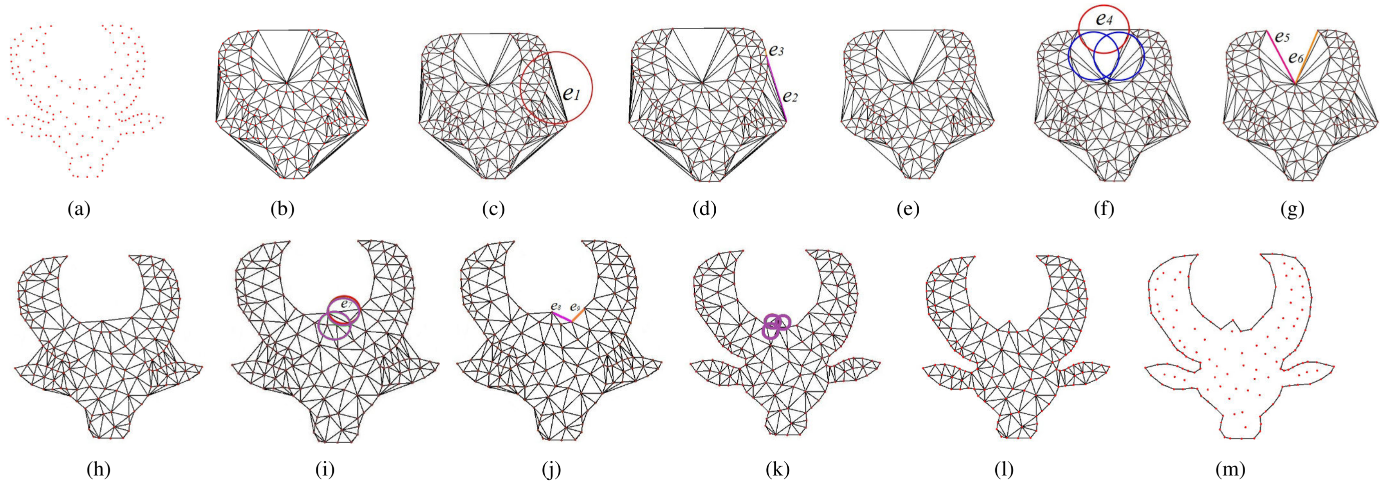


Figure 3: (a) Dot pattern, (b) DT, (c) Diametric circle of e_1, (d) e_2 and e_3 are added to the queue, (e) Intermediate graph, (f) Diametric circle of e_4 becoming midpoint circle for the adjacent sides, (g) e_5 and e_6 added to the queue, (h) Intermediate graph, (i) Empty diametric circle of e_7 and chord circles of adjacent sides (j) e_7 and e_8 added to the queue, (k) Empty diametric circles and four chord circles, (l) Final graph, (k) ec-shape

corresponding triangles as ETs) and added to the queue. G is updated to $G - e_4$. The algorithm continues further (Figure 3(h)). Figure 3(i) shows an exterior edge e_7, whose diametric circle is empty, whereas at least one of the chord circles is not empty. e_7 is removed as $G - e_7$ is regular and the queue and G are updated (Figure 3(j)). Figure 3(k) shows an EE where all the circles are empty and hence this edge cannot be removed. Figure 3(l) shows a graph when Algorithm 1 terminates. The exterior edges of G form the ec-shape (Figure 3(m)).

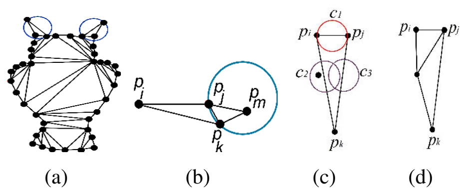


Figure 4: (a) Non-empty Diametric circles (b) Non-empty chord circle (c) Presence of a point which makes the DT invalid (d) Valid DT

4. Theoretical Guarantees

For RBS, we assume that input point set S is sampled from a polygonal object O using a modified version of $(r, \uparrow)$ sampling specified in [4]. We refer the sampling as r-sampling which is defined as follows:

DEFINITION 7 In RBS, an input point set S is sampled from a polygonal object O under r-sampling if it satisfies the following constraints:

- Each pair of adjacent boundary samples lies at a distance of at most $2r$.
- Any pair of non-adjacent boundary samples lies at a minimum distance of $2r$.

LEMMA 4.1 In RBS, assuming the input point set S is sampled from a polygonal object O using r-sampling, Algorithm 1 removes the exterior edges that are not boundary edges.

Proof Consider an ET $\triangle_{ijk} \in DT$ between three points p_i,p_j and p_k. Let d_{ij} denote $||p_i$-$p_j||$. Assume exterior edge (p_i,p_j) is not a boundary edge. Three cases are to be considered for the proof: Case 1 - $d_{ij} > 2r$, $d_{ik} < 2r$ and $d_{jk} < 2r$: In this case, the diametric circle of (p_i,p_j) is non-empty. Hence the removal of (p_i,p_j) is valid. Case 2- $d_{ij} > 2r$, $d_{ik} < 2r$ and $d_{jk} > 2r$: If the diametric circle of (p_i,p_j) is non-empty, then removal of (p_i,p_j) is valid. Otherwise, chord circle or midpoint circle of any of the other two edges is non-empty. If it is not, by the presence of a point outside all the circles, the DT is invalid. Hence non-emptiness of any of the circles implies removal of (p_i,p_j) is valid. Case 3- $d_{ij} > 2r$, $d_{ik} > 2r$ and $d_{jk} > 2r$: If (p_i,p_j) is longer than any of the other edges of ET, either of the three circles is non-empty. Otherwise, it reduces to case 2 of the proof. □

LEMMA 4.2 In RBS, assuming the input point set S is sampled from a polygonal object O using r-sampling, Algorithm 1 does not remove any of the boundary edges.

Proof Consider an ET $\triangle_{ijk} \in DT$ between three points p_i,p_j, and p_k. Assume (p_i,p_j) is a boundary edge. If a diametric circle of (p_i,p_j) is non-empty, ET has the other two edges also as boundary edges (Figure 4(a)) and it violates r-sampling. A diametric circle is empty and at least one chord circle is non-empty (Figure 4(b)) results in a polygon which is not simple, which contradicts the fact that ec-shape is a simple polygon. Both diametric and chord circles are empty and midpoint circle is non-empty (Figure 4(c))implies an invalid DT whose valid DT is shown in Figure 4(d). Hence, Algorithm 1 does not remove any of the boundary edges. □

COROLLARY 4.3 Following Lemmas 4.1 and 4.2, ec-shape is homeomorphic to a simple closed curve.

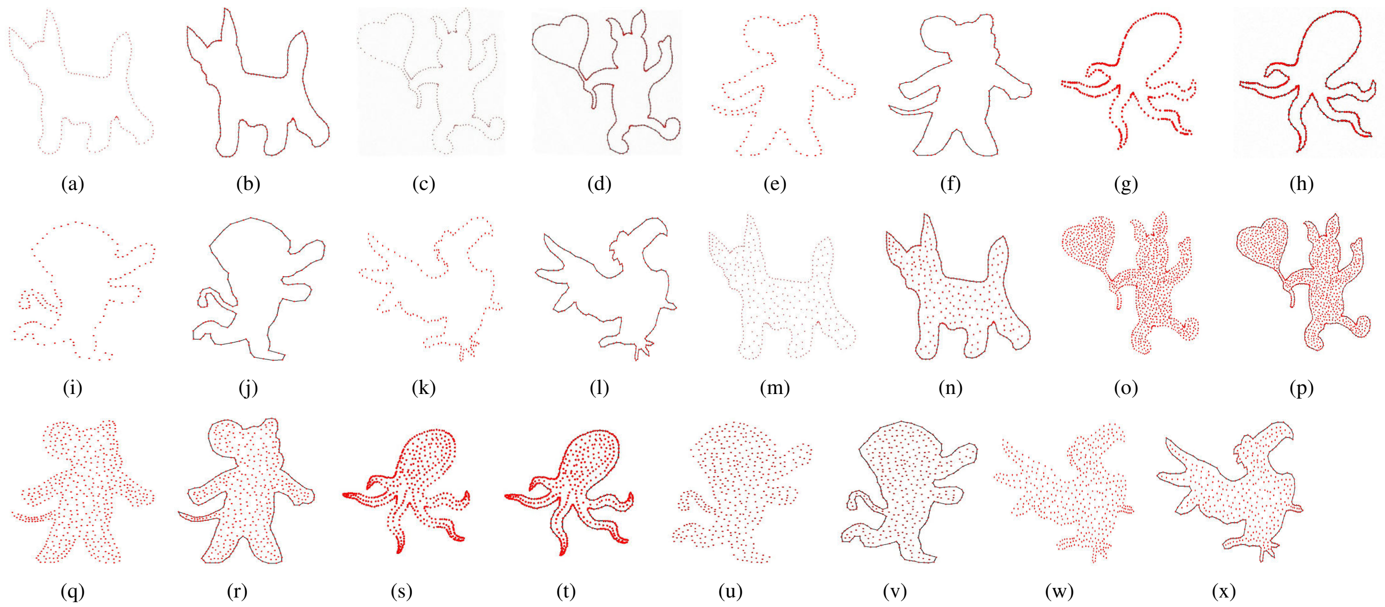


Figure 5: (a)-(l) Boundary sample and *ec*-shape pairs for different inputs, (m)-(x) Dot pattern and *ec*-shape pairs for different inputs.

Proof The non-boundary edges are removed due to Lemma 4.1 and the boundary edges are retained due to Lemma 4.2, hence *ec*-shape captures linear approximation of the original boundary and is homeomorphic to a simple closed curve. □

Due to the existence of interior points in the dot pattern, Definition 7 is modified by adding an additional constraint - *A boundary sample is at a minimum distance of 2r with respect to any non-boundary sample*. Topological guarantee for RDP can be proved in a similar way as it has been proved for RBS.

5. Results and Discussion

We implemented our algorithm in C++ using Delaunay Triangulation package and other geometric predicates available in Computational Geometric Algorithms Library [17]. It has to be noted that all the input point sets we have used for generating results and comparison purposes are generic in nature and do not follow any sampling model.

Usually, in the area of reconstruction, the theoretical guarantee is provided under certain sampling models (see Section 4 for our sampling model). Nevertheless, in practise, such sampling models are rarely achievable [12] and hence it is important to establish that an algorithm performs on generic inputs, independent of the sampling models. The input point sets used in the paper are of varying point densities and distributions and not particular to any sampling model. Few inputs and outputs of the algorithm for both boundary samples and dot patterns are as shown in Figure 5. Our results in Figure 5 clearly points out that the algorithm can handle shapes with sharp features (ears of the animal shapes in Figures 5(b) and 5(n)), non-directed boundary sample [4] (left up part in Figures 5(d) and 5(p)), elongated regions (tail of the animal shapes in Figures 5(f), 5(r), 5(j) and 5(v)), thin projections (feet of bird shapes in Figures 5(l) and 5(x)), smooth curves (upper part in Figures 5(h) and 5(t)). Figure 5 demonstrates that the algorithm can handle wide variety of shapes irrespective of the type of input point set.

5.1. Comparison with existing methods

We performed both qualitative and quantitative comparisons with the existing methods for both RBS and RDP. The existing methods we considered are Crust [10], NN-crust [11], α-shape [5], simple-shape [3], *RGG* for directed boundary sample [4] and χ-shape [9].

Crust and NN-crust are algorithms designed for curve reconstruction. Simple-shape algorithm is a unified approach for RBS and RDP. Algorithm in [4] is for RDP under directed boundary samples. χ-shape algorithm uses *DT* for RDP (and hence amenable for RBS as well). We restricted our comparison to Delaunay-based methods because of the following reasons: (i) There are quite a few proven approaches whose codes are accessible and work in two-dimensions and (ii) The implementation of the recent non Delaunay approaches does not seem to be available for two-dimensional reconstruction (such as [18] and [19], even though their 3D versions are available and working).

5.1.1. Qualitative comparison

For RBS, Figure 6 shows the comparison of *ec*-shape with crust, NN-crust, α-shape, simple-shape and χ-shape. The first column of Figure 6 shows the boundary samples. From the outputs of crust and NN-crust algorithms (second and third columns of Figure 6), it can be observed that the outputs are not closed curves. Even when closed, it need not be a simple polygon (please see left down corner in Figure 6(p)). The α-shape, simple-shape and χ-shape (4th, 5th and 6th columns of Figure 6) show that the concavities of the input boundary samples have not been captured well (even after parameter tuning),

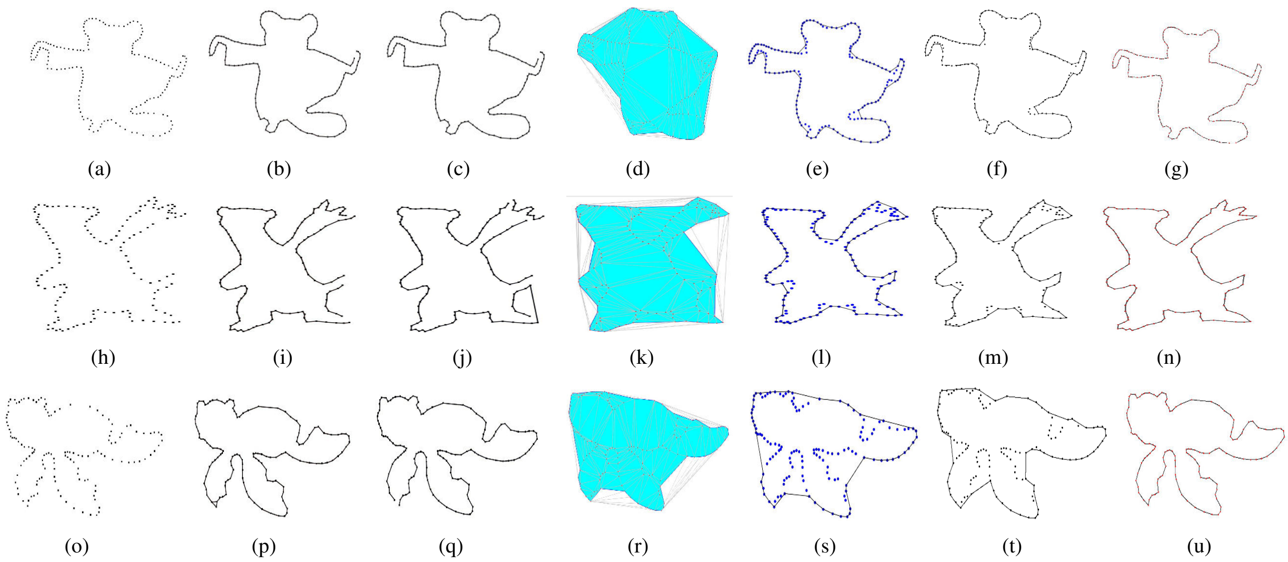


Figure 6: Inputs and outputs of RBS: 1st column - point set, 2nd - output of Crust algorithm, 3rd - output of NN-crust algorithm, 4th - α-shape, 5th - Simple-shape, 6th - χ-chape and 7th -*ec*-shape.

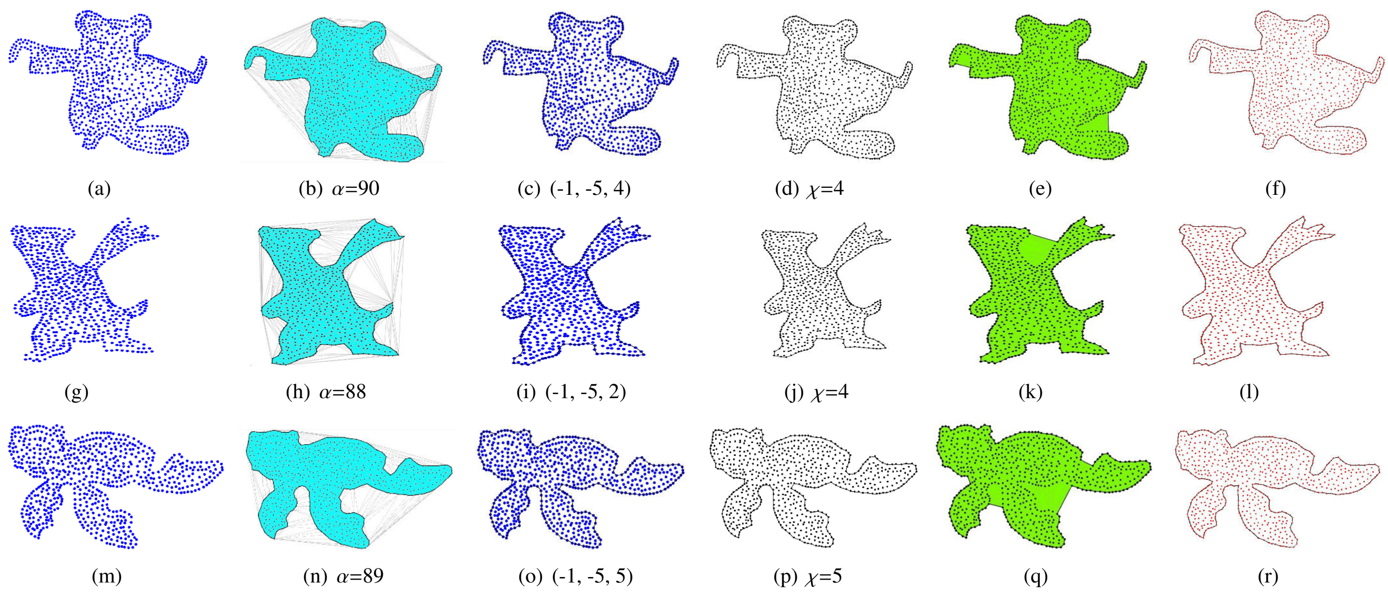


Figure 7: Inputs and outputs of RDP: 1st column - point set, 2nd - α-shape, 3rd - Simple-shape with parameters (pr_1,pr_2,pr_3), 4th - χ-shape, 5th - output of [4] and 6th - *ec*-shape.

compared to *ec*-shapes (without any parameter) shown in 7th column of Figure 6.

For RDP, we compare *ec*-shape with α-shape, simple-shape, χ-shape and output of [4]. We are not comparing *ec*-shape with the outputs of crust and NN-crust algorithms as they have been designed for RBS. Figures 7(a), 7(g) and 7(m) show the dot pattern on which RDP is performed. The fingers in the object are captured well by *ec*-shape (Figure 7(l)) than outputs of other methods (Figures 7(h)-7(k)). In the case of shape induced by the dot pattern of Figure 7(m), *ec*-shape (Figure 7(r)) performs equally well as α-shape, simple-shape and χ-shape (Figures 7(n)-7(p)) and better than output of [4] (Figure 7(q)).

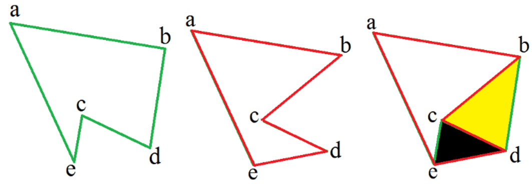
5.1.2. Quantitative comparison

In this section, experimentations on how the resultant shape varies with density and distribution of the point set have been performed and comparison with existing methods has also been discussed.

For RDP, in both density and distribution cases, we performed a quantitative comparison of *ec*-shape with α-shape, simple-shape and χ-shape, by plotting point density Vs L^2 error norm [9].

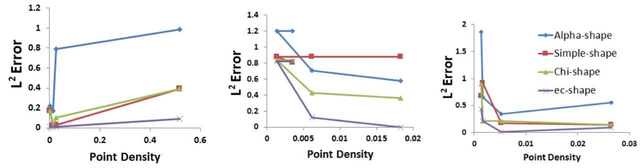
$$L^2\text{error norm} = \frac{area((O - Re) \cup (Re - O))}{area(O)}$$

where O and Re are original and reconstructed shapes respec-



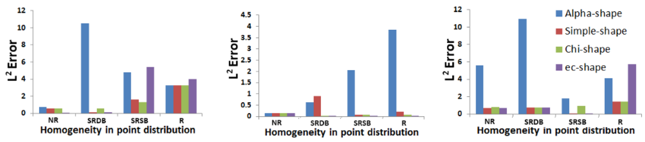
(a) (b) (c)

Figure 8: (a) Original shape (b) Reconstructed Shape (c) Symmetric difference between original and reconstructed shapes



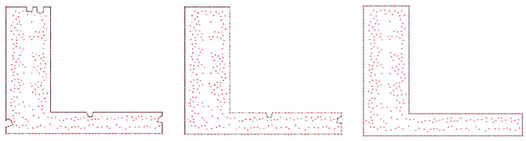
(a) (b) (c)

Figure 10: Illustration of performance of RDP in different point densities : (a) Plot for F-shape (b) Plot for G-shape (c) Plot for f-shape



(a) (b) (c)

Figure 11: Illustration of performance of RDP in different point distributions : (a) Plot for a-shape (b) Plot for L-shape (c) Plot for S-shape



(a) u=0.9 (b) u=0.8 (c) u=0.7

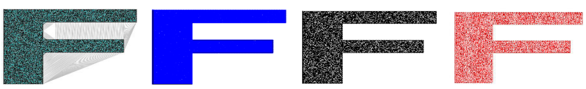
Figure 12: Illustration of parameter tuning for *ec*-shape for random point distribution

tively and – operator denotes the set theoretic difference. Given a point set, O, Re and their symmetric difference(colored region) is shown in Figure 8.

Figure 9 shows the results for F shape with different point densities for α, simple, χ (best shape obtained on visual inspection after tuning parameters) and *ec*-shapes. From the plots shown in Figure 10, it can be noticed that L^2 error norm is less in the case of *ec*-shape, compared to other shapes, illustrating that our approach performs better than the existing approaches for input point sets having varying densities.

The number of sharp corners between the two straight lines in the shape of alphabet F is more compared to other examples of G and f shapes taken for experimentation. When the point density increases the length of the edges of the *DT* formed in those sharp corners decreases. The lesser length edges of *DT* are removed later compared to longer edges and the sharp corners are not captured well when the point density increases and hence the L^2 error norm increases with increase in point density in the case of plot of the alphabet F, where as in the plots of alphabets of G and f, the error norm decreases with increase in the point density.

To experiment on how variation in point distribution affects *ec*-shape, we took four cases of point distribution: (i) nonrandom(NR), where all the points are of fixed distance from each other (ii) semi random dense boundary(SRDB), where the points are semi-randomly distributed [9] and boundary is dense (iii) semi random sparse boundary (SRSB), where the points are semi-randomly distributed and boundary is sparse and (iv) random (R), where all the points are randomly distributed. L^2 error plot for α, simple, χ and *ec* shapes for the alphabets a, L, and S shapes are shown in Figure 11.

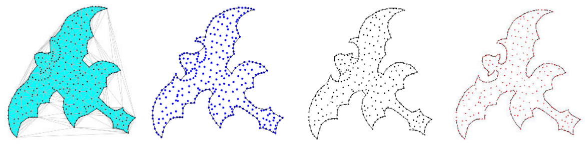


(a) α=69 (b) (-4, -5, 1) (c) χ=1 (d)

Figure 9: F-shape with point density 0.02656: 1st column - α-shape, 2nd - Simple-shape with parameters (pr_1, pr_2, pr_3), 3rd - χ-shape, and 4th - *ec*-shape.

We observe that, in the cases of SRSB and random distributions, our algorithm does not perform very well, in general, (reconstructed shape from a random distribution is shown in Figure 12(a)), which is not the case for NR and SRDB. We introduced a parameter u for diametric circle (i.e., $u * diameter$, $u \in [0, 1]$). We observed that the results improved quite a bit (Figures 12(b) and 12(c)) by tuning the parameter u for random. Plots in Figure 11, which are obtained after employing parameter tuning for the respective shapes in SRSB and random distributions, essentially show that our algorithm performs better or comparable with other algorithms. Overall, error measure from Figure 11 suggests that, in the case of nonrandom and semi-random dense boundary point distributions, *ec*-shape (without any parameter tuning) detects the boundaries with equal or less error with other parametric methods, where as in the cases of semi-random sparse boundary and random point distributions, the performance of *ec*-shape (with parameter tuning) is comparable with outputs of other parametric methods. Figure 13 illustrates the fact that even for sparse point sets, *ec*-shape captures the boundary better than the other methods.

Simple Shape Algorithm (SSA) is a parametric method with no proven topological guarantee where as *ec*-shape algorithm is non-parametric except in sparse sampling, with proven topological guarantee. Termination condition of SSA [3] depends on the input type, but *ec*-shape algorithm has a common termination condition for any input type. Algorithm in [4] is defined for reconstruction of dot patterns only where as *ec*-shape algorithm is a unified method for both dot patterns and boundary samples. As illustrated in 5th and 6th columns of Figure 7, non-directed boundary sample is captured well by our algorithm, but not by the algorithm in [4]. Please refer shape of alphabet F (Figure 9(d) in our paper) and that of Figure 20(a) in [4] to observe that our algorithm detects sharp features better. Topological guarantee specified for both papers differs because of the difference



(a) α=88 (b) (-4, -5, 5) (c) χ=5 (d) u=0.92

Figure 13: Illustration of results in sparse distributions: (a): α-shape, (b): Simple-shape with parameters (pr_1, pr_2, pr_3) , (c): χ-shape, (d): ec-shape.

in sculpting strategies. In Ball Pivoting Algorithm(BPA), multiple passes are needed to deal with unevenly sampled surfaces and BPA assumes samples distributed over the entire surface with a spatial frequency greater than or equal to an application specified value [15]. Theoretical guarantee under r-sampling is provided in our paper where as no guarantee is provided in [7].

Even though our algorithm is a unified one for reconstruction of boundary samples and dot patterns and is able to detect many prominent features of the shape induced by the input point set, it has a few limitations:

- Parameter tuning is required for detecting the boundary if the input point set is very sparse.
- Our algorithm is not capable of detecting open curves.
- Approaches using DT have the inherent disadvantage that noisy inputs cannot be handled. Our algorithm also suffers from the same.

6. Conclusions and Future Work

We have developed a unified algorithm for reconstruction of boundary samples as well as dot patterns in the plane as opposed to dealing with them separately. This approach was made possible because of the use of DT. The algorithm is simple and easy to implement with a time complexity of $O(n \log n)$ and space complexity of $O(n)$. The experimental results indicate that our algorithm is capable of detecting a wide variety of shapes having features such as sharp corners, concavities, thin regions etc. It is evident that our algorithm performs better than other approaches when the input data is not random or sparse, without the need to tune any external parameter. We have also proposed a parameter-based approach to handle very sparse and random data, which has shown to perform comparably in some cases (as well as better in few others) in comparison with existing parameter-based approaches. We have provided theoretical guarantee for reconstruction based on r-sampling. In practise, based on the extensive comparative study with existing approaches, it can be observed that ec-shape approximates the original shape quite well, independent of the sampling of the input point set.

One of the future works under consideration is island detection in both reconstruction of boundary samples and dot patterns. It also remains to be seen if the approach can be modified to handle random/sparse data without parameter tuning. Another pointer towards future work is the extension of our algorithm to three dimensions. One of the possibilities to extend the reconstruction to 3D is by using the circumsphere of an exterior facet instead of using diametric circle in 2D. In this direction, we are investigating on the modified circle and regularity constraints which might handle the removal of exterior facets of an exterior tetrahedron.

Acknowledgement

Authors would like to thank all the anonymous reviewers whose comments have helped immensely in improving the quality of the paper. We also like to thank Ministry of Human Resource Development, Government of India for the financial assistance.

References

[1] H. Edelsbrunner, Shape reconstruction with delaunay complex., in: C. L. Lucchesi, A. V. Moura (Eds.), LATIN, Vol. 1380 of Lecture Notes in Computer Science, Springer, 1998, pp. 119–132.

[2] A. Galton, M. Duckham, What is the region occupied by a set of points?, in: M. Raubal, H. J. Miller, A. U. Frank, M. F. Goodchild (Eds.), GIScience, Vol. 4197 of Lecture Notes in Computer Science, Springer, 2006, pp. 81–98.

[3] A. Gheibi, M. Davoodi, A. Javad, F. Panahi, M. M. Aghdam, M. Asgaripour, A. Mohades, Polygonal shape reconstruction in the plane, IET computer vision 5 (2) (2011) 97–106.

[4] J. Peethambaran, R. Muthuganapathy, A non-parametric approach to shape reconstruction from planar point sets through delaunay filtering, Computer-Aided Design 62 (0) (2015) 164 – 175.

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

[6] J.-D. Boissonnat, Geometric structures for three-dimensional shape representation, ACM Trans. Graph. 3 (4) (1984) 266–286.

[7] M. Attene, M. Spagnuolo, Automatic surface reconstruction from point sets in space, Comput. Graph. Forum 19 (3) (2000) 457–465.

[8] F. de Goes, D. Cohen-Steiner, P. Alliez, M. Desbrun, An optimal transport approach to robust reconstruction and simplification of 2d shapes., Comput. Graph. Forum 30 (2011) 1593–1602.

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

[10] N. Amenta, M. Bern, M. Kamvysselis, A new voronoi-based surface reconstruction algorithm, in: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '98, ACM, New York, NY, USA, 1998, pp. 415–421.

[11] T. K. Dey, P. Kumar, A simple provable algorithm for curve reconstruction, in: SODA'99, 1999, pp. 893–894.

[12] N. Amenta, S. Choi, R. K. Kolluri, The power crust, in: Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications, SMA '01, ACM, New York, NY, USA, 2001, pp. 249–266.

[13] S. Petitjean, E. Boyer, Regular and non-regular point sets: Properties and reconstruction, Computational Geometry 19 (2¢3) (2001) 101 – 126, combinatorial Curves and Surfaces.

[14] R. C. Veltkamp, Closed Object Boundaries from Scattered Points, Vol. 885 of Lecture Notes in Computer Science, Springer, 1994.

[15] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, G. Taubin, The ball-pivoting algorithm for surface reconstruction, IEEE Transactions on Visualization and Computer Graphics 5 (1999) 349–359.

[16] B. Bollobas, Modern Graph Theory, Springer, 1998.

[17] CGAL, Computational Geometry Algorithms Library, http://www.cgal.org.

[18] M. Kazhdan, M. Bolitho, H. Hoppe, Poisson surface reconstruction, in: Proceedings of the Fourth Eurographics Symposium on Geometry Processing, SGP '06, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2006, pp. 61–70.

[19] M. M. Kazhdan, H. Hoppe, Screened poisson surface reconstruction, ACM Trans. Graph. 32 (3) (2013) 29.