Characterizing the shapes of noisy, non-uniform, and disconnected point clusters in the plane

Xu Zhong,⁎, Matt Duckham

Department of Infrastructure Engineering, University of Melbourne, Victoria 3010, Australia
School of Mathematical and Geospatial Sciences, RMIT University, Victoria 3001, Australia

A R T I C L E   I N F O

Article history:
Received 6 December 2015
Received in revised form 12 January 2016
Accepted 16 January 2016
Available online 3 February 2016

Keywords:
Footprint
Non-convex
Delaunay triangulation
Clustering
χ-Shape
DBSCAN

A B S T R A C T

Many spatial analyses involve constructing possibly non-convex polygons, also called “footprints,” that characterize the shape of a set of points in the plane. In cases where the point set contains pronounced clusters and outliers, footprints consisting of disconnected shapes and excluding outliers are desirable. This paper develops and tests a new algorithm for generating such possibly disconnected shapes from clustered points with outliers. The algorithm is called χ-outline, and is based on an extension of the established χ-shape algorithm. The χ-outline algorithm is simple, flexible, and as efficient as the most widely used alternatives, O(nlogn) time complexity. Compared with other footprint algorithms, the χ-outline algorithm requires fewer parameters than two-step clustering-footprint generation and is not limited to simple connected polygons, a limitation of χ-shapes. Further, experimental comparison with leading alternatives demonstrates that χ-outlines match or exceed the accuracy of χ-shapes or two-step clustering-footprint generation, and is more robust to some forms of non-uniform point densities. The effectiveness of the algorithm is demonstrated through the case study of recovering the complex and disconnected boundary of a wildfire from crowdsourced wildfire reports.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

Many real-life applications require the construction of polygonal regions that characterize the distribution of a set of points in the plane PC2 (e.g., Downs and Horner (2009)). Such regions are called “footprints” of P. A typical structure used to generate a footprint is the convex hull. The convex hull of P is the smallest convex polygon that contains all points in P (De Berg, Van Kreveld, Overmars, & Schwarzkopf, 2000). However, in cases where the distribution of points is markedly non-convex, there can be no single “correct” footprint. Rather, the accuracy of a footprint may depend on the specific application, or on human cognition and preference. Further, the point distribution may be best characterized by disconnected polygons, possibly neglecting outliers (Lee & Estivill-Castro, 2006; Lee, Qu, & Lee, 2012). Fig. 1 shows an example of a point set P that contains pronounced clusters and outliers. In such a case the convex hull significantly fails to capture the shape of P, illustrated in Fig. 1(a).

Today, several algorithms exist to construct “accurate” footprints for such non-convex and clustered point distributions. Because there can be no unique non-convex polygon, such algorithms require at least one adjustable parameter to obtain desirable footprints. This paper develops and tests a new algorithm for generating possibly disconnected polygons that characterize the shapes of such non-convex and clustered point distributions. The algorithm, called χ-outline is an adaptation of the established χ-shape algorithm (Duckham, Kulik, Worboys, & Galton, 2008) to handle disconnected shapes and outliers using only a single parameter. Fig. 1(b) illustrates a typical output of the algorithm.

Following a review of the background literature in Sections 2 and 3 describes the χ-outline algorithm itself in detail. Section 4 then evaluates the performance of the algorithm against the two leading alternatives: χ-shapes, and a two-step clustering-footprint generation based on DBSCAN and χ-shapes. The evaluation shows that in most cases the accuracy of χ-outlines equals or outperforms these alternatives, in particular where individual clusters tend to differ in point densities. Section 5 illustrates the application of χ-outlines to a case study of wildfire perimeter estimation based on crowdsourced fire reports, where systematic differences in cluster densities are common, due to variations in population density. Finally, Sections 6 and 7 provide a discussion of the results and the final conclusions, respectively.

2. Background

This paper focuses on footprint algorithms where the goal is to construct a polygonal region that adequately represents the distribution of a given set of points, P, that occupies one or more regions in the plane. We do not here consider the special case where points in P lie only on curves or the boundary of regions. Numerous algorithms already exist to generate curves or outlines for this latter case (e.g. Amenta, Choi, & Kolluri, 2001a, 2001b, Attali, 1997, Traka & Tziritas, 2003).
2.1. Simple region-based footprints

One characteristic frequently used to distinguish footprint algorithms is whether or not all the input points in \( P \) are required to lie in the interior of the footprint. For example, covering discs and the covering discs with tangents methods (Galton & Duckham, 2006) assign an influence region to each point in \( P \). The footprint is the union of the influenced regions of all points in \( P \). The shape and size of the influenced region decide the resulting footprint, but all input points in \( P \) will be contained in the footprint.

The Voronoi diagram (VD) based method of Alani, Jones, & Tudhope (2001) and the Delaunay triangulation (DT) based method of Arampatzis et al. (2006) also produce footprints that contain all the input points. These methods additionally require auxiliary points \( P_a \) which are considered to be outside of the footprint. The principle is to construct a footprint whose edges separate a point in \( P \) from its neighboring points in \( P_a \), so that the footprint contains all points in \( P \) and excludes all points in \( P_a \). The edges of the VD and the DT of \( P \cup P_a \) are used to generate the edges of the footprint. In the VD based method, Voronoi edges that are shared by a Voronoi cell of a point in \( P \) and a Voronoi cell of a point in \( P_a \) form the footprint (Alani et al., 2001). In the DT based method, the edges of the footprint are generated by connecting the midpoints of the triangulation edges that connect a point in \( P \) and a point in \( P_a \) (Arampatzis et al., 2006). These two methods are most appropriate in cases where \( P_a \) is given, i.e., where the problem involves a set of both positive (included) and negative (excluded) points. Galton and Duckham (2006) described an iterative approach to use these two methods when \( P_a \) is not given. An initial \( P_a \) is generated randomly outside the convex hull of \( P \) to produce a preliminary footprint. Then a new \( P_a \) is generated outside the preliminary footprint, which produces a new footprint. The final footprint can be obtained by a pre-defined number of successive iterations.

In many cases, footprint algorithms generate a single, simple polygon to characterize the shape of \( P \). For example, the \( k \)-nearest neighbors (kNN) based method (Moreira & Santos, 2007) was proposed by generalizing the gift-wrapping convex hull algorithm (Jarvis, 1973). At each iteration, the kNN-based algorithm finds the next point from the \( k \)NN of current points, instead of processing the entirety of \( P \) used in the gift-wrapping algorithm. Each subsequent point produces the largest right-hand turn from the current point without resulting in self-intersection. When no legal subsequent points exist, the algorithm has to increase \( k \) and rerun from the beginning. The algorithm also increases \( k \) and reruns from the beginning if it generates a footprint that does not include all points in \( P \). Hence the algorithm has a poor worst-case efficiency. The \( \chi \)-shape algorithm (introduced in detail in Section 3) proposed by Duckham et al. (2008) is more efficient than the kNN-based method, but also can only produce a single, simple polygonal footprint.

2.2. Pre-clustering and outliers

To characterize more complicated and clustered point distributions, such as illustrated in Fig. 1, input points may be pre-processed using a spatial clustering algorithm (e.g., Miller & Han, 2009, Xu & Wunsch, 2005). In general, clustering may be able to handle a priori unknown numbers of clusters with arbitrary shapes, as well as outliers. For example, one of the most widely used spatial clustering algorithms is the density-based DBSCAN (Ester, Kriegel, Sander, & Xu, 1996). Clustering using DBSCAN can partition the point set \( P \) into one or more disjoint clusters \( P_i \) and possibly a set of outliers \( O \). After clustering, any of the footprint algorithms discussed above, including the kNN-based algorithm or the \( \chi \)-shape algorithm, can be applied to each cluster independently.

The approach of pre-clustering process allows independent identification of clusters and outliers, providing great flexibility in footprint generation. But there are two disadvantages. First, any pre-clustering algorithm necessarily requires additional parameterization. Just as for non-convex footprint generation, there can be no single correct answer for clustering. Selecting the correct clustering parameter may be difficult to achieve automatically. The quality of the constructed footprints is then strongly dependent on the parameterization of the clustering algorithm. Second, depending on the footprint algorithm parameterization, the footprints of clusters may intersect with one other. In this case, the union of the footprints needs to be calculated to obtain regular polygonal shapes. In addition, the union of the footprints possibly contains holes, which also need to be detected and removed for regular polygonal shapes. These additional steps may also increase the complexity and computational overhead of this approach.

2.3. Footprints for clusters and outliers

Some well-known footprint algorithms do enable the construction directly of disconnected polygonal shapes for points with clustered distributions. The output footprints of these algorithms may consist of polygons, lines, and isolated points. The polygonal shapes may not contain all points in \( P \). These algorithms can separate sampling points (points contained in the polygonal shapes) from outliers (points not contained in the polygonal shapes) by themselves, without the need of pre-clustering. The \( \alpha \)-shape algorithm (Edelsbrunner, Kirkpatrick, & Seidel, 1983) is the most famous example, which generalizes the convex hull with a single parameter \( \alpha \), the multiplicative inverse of the radius of closed disks. Negative parameters yield the complement of an open disk of radius \(-1/\alpha\). The \( \alpha \)-shape of \( P \) is a sub-graph of the DT of \( P \). For large negative \( \alpha \), the \( \alpha \)-shape of \( P \) is just \( P \) itself. When \( \alpha = 0 \), the \( \alpha \)-shape is the convex hull of \( P \).
The \( \alpha \)-shape (Melkemi, 1997; Melkemi & Djebali, 2000) contains the \( \alpha \)-shapes, but uses a different parameterization method. Similar to the VD based and the DT based footprints algorithms above, the \( \alpha \)-shape takes a set of auxiliary points \( P_{\alpha} \) in addition to the parameterization. The VD of \( PU_P \) is firstly computed. Then any pair of points \( p, q \in P \) whose Voronoi cells are adjacent to each other and to a common Voronoi cell of a point in \( P_{\alpha} \) are connected, forming the \( \alpha \)-shape. Thus the \( \alpha \)-shape is a sub-graph of the DT of \( P \). However, as for the VD and DT methods discussed above, this is most appropriate in scenarios where the negative points are already given.

Chaudhuri, Chaudhuri, and Parui (1997) proposed two footprints—\( r \)-shapes and \( s \)-shapes—which, unlike \( \alpha \)-shapes and \( \alpha \)-shapes, are not based on a pre-defined graph (VD or DT) of \( P \). In the \( \alpha \)-shape algorithm, the plane is pixelated first, i.e., partitioned into a grid of square cells of side-length \( s \). Then \( s \)-shapes are constructed by assembling the grid cells containing points in \( P \). The \( r \)-shape algorithm is related to the covering discs method discussed above. All the points in \( P \) are assigned to a disk of radius \( r \). Any pair of points \( p, q \in P \) is connected if the boundary of their disks and the boundary of the union of all the disks intersect at one point. Such edges form the \( r \)-shape. Constructing \( r \)-shapes and \( s \)-shapes takes \( O(n^2) \) time where \( n \) is the number of points in \( P \), more efficient than VD based or DT based footprints which require at least \( O(n \log n) \) time.

Garai and Chaudhuri (1999) developed a “split and merge” procedure to construct possibly disconnected footprints based on the convex hull of \( P \). The splitting step successively inserts extra edges to carve off pieces of the convex hull to approach the perceived pattern of \( P \). Then the isolation step separates the components in the pattern of \( P \). The outline resulting from these repeated splitting and isolation steps can be highly zigzagged, but is made smoother by the final merge step. A unique feature of this algorithm is that one can specify the number of edges in the output footprint.

The close-pairs method (Galton & Duckham, 2006) connects any pair of points \( p, q \in P \) if the distance from \( p \) to \( q \) is not greater than some threshold distance \( r \). The close-pairs footprint of \( P \) is constructed by the outline of the edges between all connected pairs. The vertices of the footprint may contain points that are not in \( P \). The close-pairs method is computationally expensive, having a time complexity \( O(n^2) \). The output footprints are not guaranteed to be regular.

Despite this wealth of existing algorithms, this paper presents a new \( \chi \)-shape-based footprint algorithm, called \( \chi \)-outlines, that fills a gap in this previous work. First, the \( \chi \)-outline algorithm is as efficient (\( O(n \log n) \) time complexity) as the most widely-used single-step algorithms, \( \alpha \)-shapes and \( \chi \)-shapes. Second, the \( \chi \)-outline algorithm can generate multiple disconnected footprints and excluding outliers using a single parameter. This obviates any need for multi-step or multi-parameter clustering required by some simple approaches. Further, unlike \( \alpha \)-shapes, the individual polygon components of a \( \chi \)-outline are guaranteed to be simple and regular, containing no holes or islands. Finally, as we shall see, the adaptive nature of the \( \chi \)-outline makes it especially tolerant to clusters with differing point densities, more so than \( \alpha \)-shapes for instance. Such systematic changes in point densities are not uncommon in real-world spatial data, such as the systematic effects of population density upon crowdsourced spatial data.

3. \( \chi \)-Outline algorithm

Our \( \chi \)-outline algorithm is based on the established \( \chi \)-shape algorithm. \( \chi \)-shapes are based on the DT (Delaunay triangulation) of input points. The principle of the \( \chi \)-shape algorithm is to remove exterior edges of the triangulation of \( P \) in descending order of length, subject to a regularity constraint. The regularity constraint of \( \chi \)-shapes requires that at all times the exterior edges of the triangulation form the boundary of a simple (Jordan) polygon (the \( \chi \)-shape itself). Every time an exterior edge is marked for removal, the polygon formed by the new set of exterior triangulation edges is checked for topological irregularities to the polygon (i.e., namely isolated points or lines or disconnected components).

Based on this distinction between regular and irregular polygons formed by the exterior edges of a triangulation, we can define two types of exterior edges:

**Definition 1.** An exterior edge of a triangulation is termed **irregular** if removing that edge would result in the exterior edges of the resulting triangulation forming an irregular polygon. Otherwise an exterior edge of a triangulation is termed **regular**.

The polygon formed by the exterior edges of the triangulation at any iteration of the \( \chi \)-shape algorithm is guaranteed to be a simple polygon that contains all the input points. The algorithm ceases if there is no regular exterior edge longer than the length threshold \( \lambda \), which is derived from a single normalized length parameter \( \chi \). From [0,1]. \( \chi \) is converted to \( \lambda \) linearly as:

\[
\lambda = \chi \left( \max(DT_b) - \min(DT_e) \right) + \min(DT_e),
\]

where \( DT_b \) and \( DT_e \) are the length of exterior edges and all edges of DT, respectively. When \( \chi = 1 \), the \( \chi \)-shape is the convex hull of the input points. When \( \chi = 0 \), the \( \chi \)-shape is a uniquely defined connected polygon with minimum area.

As discussed in Section 1, in cases where points contain distinct clusters of points or outliers, the \( \chi \)-shape is not appropriate. Instead, input points would need to be pre-processed with a spatial clustering algorithm before calculating \( \chi \)-shapes on each of the clusters individually. While this approach has the advantage of modularity, reusing two existing algorithms, it also increases both the difficulty of finding a good parameterization (one must parameterize both the clustering and the \( \chi \)-shape algorithms) as well as the computational efficiency of the whole process. In order to address these disadvantages, the \( \chi \)-outline algorithm extends \( \chi \)-shapes to allow disconnected shapes and outlier elimination, but still using a single parameter.

3.1. Extending the \( \chi \)-shape algorithm

An obvious modification to the \( \chi \)-shape algorithm would be to remove the regularity constraint, and remove edges of the triangulation that are irregular. However, such a modification tends to lead to highly undesirable outputs that are neither regular polygons nor adequately capture the shape of the input point set.

Instead, the \( \chi \)-outline algorithm relaxes the regularity constraint to allow irregular exterior edges to be removed in a way that maintains the regularity of the output polygons. This is based on the observation that there is a high likelihood that exterior edges that are irregular may often connect one cluster to another (e.g., Fig. 2(a)) or to an outlier (e.g., Fig. 2(c)), or connect an outlier to another outlier (e.g., Fig. 2(b)). By identifying these circumstances, removing such exterior edges splits the polygon that contains multiple clusters (e.g., Fig. 2(a)), or isolates outliers (e.g., Figs. 2(b)–(c)).

Algorithm 1 presents the pseudo code of the \( \chi \)-outline algorithm. In explaining in natural language the steps the \( \chi \)-outline uses to determine when and how to remove exterior edges that are irregular the following definitions help with conciseness.

**Definition 2.** The **adjacent triangles** of an edge \( e \) in a triangulation is the set of one (if \( e \) is an exterior edge) or two (if \( e \) is an interior edge) triangles in the triangulation that are incident with (i.e., adjacent to) \( e \).

**Definition 3.** The **arms** of an exterior edge \( e \) in a triangulation are the other two edges incident with the (unique) adjacent triangle of \( e \).

**Definition 4.** An irregular exterior edge \( e \) of a triangulation is **isolating** if either of the arms of \( e \) is also an exterior edge.
If an irregular exterior edge \( e \) of the triangulation is not isolating (e.g., Fig. 2(a)), \( e \) can be handled in the same way as regular exterior edges. Removing \( e \) generates two new exterior edges of the resulting triangulation. The exterior edges of the resulting triangulation will form two regular polygons that share a single common vertex. An isolated vertex can be isolated from the rest vertices of the graph. An isolated vertex can be identified by checking if its degree is 2 before removing \( e \). The isolated vertices can be eliminated as outliers. In addition, this process possibly separates exterior edges. If one of the arms of the isolating exterior edge \( e \) is not an exterior edge, it becomes an exterior edge in the resulting triangulation. This process may cause at most three vertices contained in the adjacent triangle of \( e \) isolated from the rest vertices of the graph. An isolated vertex can be identified by checking if its degree is 2 before removing \( e \). The isolated vertices can be eliminated as outliers. In addition, this process possibly separates one connected component into two or three connected components (excluding isolating points). The exterior edges of the connected components form multiple regular polygons, which do not intersect with each other.

The \( \chi \)-outline algorithm follows the basic course of the \( \chi \)-shape algorithm, but using the strategies above in place of the simple irregularity constraint. The \( \chi \)-outline algorithm terminates when no exterior edge is longer than the length threshold \( \lambda \). The \( \chi \)-outline generated is a set of regular polygons, some of which may possibly share a single vertex with each other. Each polygon corresponds to the footprint of one cluster. Some points (outliers) may be excluded from the footprints.

As discussed previously, this paper focuses on the case where input points occupy possibly multiple regions but not the case where points in \( P \) lie only on curves or the boundary of regions. Hence a simple heuristic is used to eliminate the polygons that contain only a small number of vertices. Let \( \chi\text{outline}(P) \) be the set of polygons obtained from the \( \chi \)-outline algorithm as described above. The output \( \chi \)-outline \( F_\chi \) of \( P \) is then the set of polygons:

\[
F_\chi(P) = \{ f | f \in \chi\text{outline}(P) \text{ and the number of points in the interior of } f \geq \lceil \ln|P| \rceil \},
\]

where \( \lfloor x \rfloor \) is the nearest integer to \( x \).

### 3.2. Adaptive parameterization

In cases where distinct clusters of points are present in \( P \), the length thresholds \( \lambda \) of the \( \chi \)-shape algorithm derived from Eq. (1) for the different clusters may be different. Instead of using a global length parameter \( \lambda \) in the \( \chi \)-outline algorithm, it is possible to allow the parameterization of \( \chi \)-outlines to adapt to the structure of the points in each cluster as new clusters are uncovered. This adaptive parameterization is expected to generate better and more flexible results compared to the basic \( \chi \)-outline algorithm described in the previous section, which uses one global length threshold for all edges. Hence the basic \( \chi \)-outline algorithm is modified further to adapt the threshold \( \lambda \) to each connected component as the algorithm cleaves off parts from the original DT of the input points.

At the commencement of the \( \chi \)-outline algorithm, all input points are in one connected component, i.e., the DT of the points. Sometimes removing an isolating exterior edge splits one connected component into two or three connected components (excluding isolating points). Any connect component that does not satisfy the criterion defined by Eq. (2) will be eliminated eventually. Hence in the adaptive \( \chi \)-outline algorithm such connected components are eliminated when they are first created. If more than one newly created connected component is retained, these are considered as new components. A new length threshold is calculated for each new component. A reasonable heuristic length threshold \( \lambda_C \) for a new connected component \( C \) is.

\[
\lambda_C = \mu_C + k_C \sigma_C,
\]

where \( \mu_C \) and \( \sigma_C \) are the mean and standard deviation of the edge length of \( C \), respectively, when \( C \) is created.

Multiple clusters or outliers in \( C \) tend to lead to a relatively large \( \sigma_C \). In these cases, a relatively small \( k_C \) should be used to remove more edges incident to two clusters or an outlier. Conversely, if most of the points in \( C \) belong to a single cluster, \( \sigma_C \) will be relatively small. A relatively large \( k_C \) should be used to retain more edges. Setting \( k_C = \chi \sigma C / \sigma_C \) satisfies these requirements, where \( \sigma \) is the mean of \( \sigma_C \) of all existing connected components. The parameter \( \chi \) ensures the flexibility of the \( \chi \)-outline algorithm, which directly controls the proportion of retained edges in the DT. The \( \chi \)-outline algorithm returns the convex hull when \( \chi \geq (\max(D_{T_b}) - H_{DT}) / H_{DT} \), where \( H_{DT} \) and \( \sigma_{DT} \) are the mean and standard deviation of the edge length of DT, respectively. Finally, the length threshold \( \lambda_C \) for the new connected component \( C \) is calculated as

\[
\lambda_C = \mu_C + \chi \sigma C / \sigma_C
\]

As the edges in different connected components have different length thresholds, the exterior edges are removed in descending order of the difference between the length and the associated length threshold of the edges. The \( \chi \)-outline algorithm stops when no exterior edge is longer than the associated length threshold. Algorithm 1 presents the pseudo code of the \( \chi \)-outline algorithm.
Algorithm 1. $\chi$-outline algorithm (pseudo code).

Input: A finite set of two-dimensional points $P$ in $\mathbb{R}^2 \times \mathbb{R}$. One parameter $\chi \in \mathbb{R}$.
Output: $\chi$-outline $F_\chi(P, \chi)$.

1. Construct the DT $\Delta$ of $P$.
2. Construct the list $B$ of exterior edges of $\Delta$.
3. Construct the list $A$ of length threshold associated to exterior edges in $B$ using Equation 4.
4. Sort the list $B$ in descending order of $\left(\text{length}(B) - \chi\right)/L$ is the edge length function.
5. While $E(\chi) \neq A(\chi)$ do
   1. If $|E(\chi)| = |A(\chi)|$ then
      1. Remove $B(\chi)$ from $A(\chi)$.
      2. Remove the arms of $B(\chi)$ that are in $A(\chi)$.
      3. Insert the arms of $B(\chi)$ that are in $B(\chi)$.
      4. If one connected component is split into multiple new connected components then
         1. Remove too small connected components using Equation 2.
      5. More than one newly created connected components are retained then
      7. Sort the list $B$ and $A$ in descending order of $(\text{length}(B) - \chi)$.
   2. end
6. end
7. end
8. Return the polygons $F_\chi$ formed by $B$.

3.3. $O(n \log n)$ time complexity

The original $\chi$-shape algorithm has a computational complexity $O(n \log n)$ where $n$ is the number of input points. The $\chi$-outline algorithm extends the $\chi$-shape algorithm by adding several steps.

The first additional computation is to determine if an irregular exterior edge $e$ is isolating. This step needs to check if the arms of $e$ are exterior edges, which can be achieved in $O(\log |B|)$ time, where $B$ is the list of exterior edges of the triangulation. In the worst case where $|B| = n$, the time complexity of this process is $O(n \log n)$. This additional step scans every edge of the DT in the worst case. Thus the total complexity is $O(n \log n)$ where $n$ is the number of edges in the DT. By Euler’s formula, the total number of edges in a planar triangulation is $O(n)$ where $n$ is the number of vertices in the planar triangulation. Hence the time complexity of this additional step is $O(n \log n)$.

Every time an isolating irregular exterior edge $e$ is removed, the edge is checked to see if any vertex of the boundary triangle containing $e$ is isolated. This step needs to query the degree of three vertices that are known; and at most three new connected components are created at a time, filtering the connected components by Eq. (2) can be done in constant time. If more than one newly created connected component is retained, it takes at most $O(\log n)$ time to calculate the length thresholds for the new connected components. The list of exterior edges $B$ needs to be sorted in ascending order of the difference between the lengths and the associated length thresholds of the exterior edges, which takes $O(n \log n)$ time. Hence the creation of new connected components fires a process of time complexity $O(n \log n)$. The creation of new connected components is occasional. The frequency depends on the number of clusters in the input data, which is much smaller than $n$ and independent on $n$. Hence this $O(n \log n)$ process runs for just a small number times with approximately $O(n \log n)$ total time complexity.

All the additional steps to the original $\chi$-shape algorithm can be done within $O(n \log n)$ time. Hence the computational complexity of the $\chi$-outline algorithm is still $O(n \log n)$. While in the worst case several of these steps may require $O(n \log n)$ time complexity, in practice we expect these steps to occur relatively infrequently and place a much lower average-case burden on the computation.

4. Experimental evaluation

The performance of the $\chi$-outline algorithm was evaluated experimentally through comparison with two of the leading state-of-the-art alternatives: $\alpha$-shapes and a two-step clustering-footprint technique.

The two-step clustering-footprint technique combined DBSCAN clustering, one of the most successful and widely used point clustering algorithms, with $\chi$-shapes footprints. This provides a direct comparison between the performance of the three-parameter DBSCAN plus $\chi$-shape approach (two parameters in DBSCAN and one parameter in $\chi$-shape), and the one-parameter $\chi$-outline. When comparing with $\alpha$-shapes, a few minor improvements were made to the outputs to ensure the most rigorous possible test of our algorithm. In general, $\alpha$-shapes may contain arbitrarily many holes and islands, as well as isolated points and other topological irregularities. To aid fair comparison with the disconnected, simple components in $\chi$-outlines and the output of the two-step DBSCAN plus $\chi$-shapes, the $\alpha$-shape output was modified by filling in holes and removing topological irregularities when they arose. Thus, the modified $\alpha$-shapes also consist of regular polygons, which possibly share a single vertex with each other (just like the $\chi$-outline).

The above modified $\alpha$-shapes may contain small polygons formed by small groups of close-by outliers. Hence the filtering criterion (Eq. (2)) was also applied to the modified $\alpha$-shapes to ensure fair comparison with the other two algorithms.

Three experiments were designed to evaluate the performance of the $\chi$-outline algorithm. The first experiment explored how the above three approaches respond to different proportions of outliers. Second, the impact of increasing inhomogeneity in point distribution upon the performance of the three approaches was studied. Finally, the effect of non-uniform cluster density upon the performance of the three approaches was also investigated.

4.1. Experimental setup

In order to compare the three algorithms quantitatively, an experiment to test the algorithms’ abilities to reconstruct shapes derived from known “ground truth” point patterns was designed. Two types of shapes were tested in this experiment. First, six hole-free letters were arbitrarily chosen for this test: “H”, “E”, “X”, “C”, “S”, and “W”, placed on a $2 \times 3$ grid. The font of letters was Arial, a sans-serif font. The letters were placed far away enough from each other such that the footprints were expected to be disconnected (see Fig. 3(a)). The second type of shapes for this test was the borders of two countries, USA and Malaysia, that consist of disconnected polygons (see Figs. 3(b)-(c)).

4.1.1. Generating point sets

The ground truth shapes were filled with 600 semi-random sample points—random points drawn from a uniform distribution with minimum allowed pair-wise distance $d = \sqrt{\frac{1}{n}}$, where $A$ is the total area of ground truth shapes; $r$ is the normalized minimum allowed pair-wise distance which is a ratio between $nd_A$ and $A$. Truly random points tend to be highly inhomogeneous, forming many clusters and holes that confound most footprint algorithms and distorting the true shape from which the points are drawn. Hence, the semi-random points
were used for the first and third experiments, where \( r = 3 \times 10^{-3} \). In the second experiment, this constraint was relaxed, and the effect of inhomogeneity in point distribution was explored through changing the normalized minimum allowed pairwise distance \( r \).

### 4.1.2. Generating outliers

Random outliers in the experiments were drawn from a uniform distribution within the study area outside the ground truth shapes (light gray region in Fig. 3). For each experimental setting, 100 independent replications of input points (semi-random sample points and purely random outliers) were drawn and processed.

### 4.1.3. Evaluation metric

The footprints of the input points were compared with the true shapes using an area-based evaluation metric (Nascimento & Marques, 2003). Area-based precision is the proportion of the true area that intersects the true shapes. Area-based recall is the proportion of the true area that is captured by the footprint. Overestimation can lead to high recall but low precision. Conversely, underestimation can yield high precision but low recall. The area-based F1-score, the harmonic mean of area-based precision and recall, was used as the accuracy score of a footprint.

### 4.1.4. Optimal parameterization

As already discussed, all footprint algorithms require at least one parameter to work as there can be no unique non-convex footprint. Our experiments examined the optimal parameterization for each algorithm to enable a fair comparison. The optimal parameterization was that which led to the highest possible area-based F1 score.

The DBSCAN clustering algorithm has two parameters: \( \epsilon \) (radius of neighborhood search) and \( minPts \) (the minimum number of neighbors of a core point). \( minPts \) can be set effectively by a simple heuristic to be the nearest integer \( \geq 2 \) to \( \ln|P| \) (Birant & Kut, 2007). As the features (sample points or outliers) of input points were known in advance in our simulation experiments, \( \epsilon \) can be optimized for the input points. We considered the DBSCAN algorithm as an information retrieval function. True positives are the sample points identified as a point in the output clusters of the DBSCAN algorithm. Then precision is the number of true positives divided by the total number of points in the output clusters. Recall is the number of true positives divided by the total number of sample points. F1-score is the harmonic mean of precision and recall. For each experimental setting, \( \epsilon \) was scanned within an adequately large and reasonable parameter space discretely with a small enough scanning interval. The \( \epsilon \) value that led to the maximum median F1-score for the 100 replications of input points was set as the optimal \( \epsilon \) for the experimental setting.

The optimal \( \chi \) value for the \( \chi \)-shape and \( \chi \)-outline algorithms, and the optimal \( \alpha \) radius \((1/\alpha)\) for the modified \( \alpha \)-shape algorithm were determined by scanning an adequately large and reasonable parameter space discretely with a small enough scanning interval. The \( \chi \) parameter of the \( \chi \)-shape algorithm was tested between 0 and 1 with an interval 0.01. \( \chi \)-outline was calculated with \( \chi \) changed from 0 to 2 with an interval 0.02. For modified \( \alpha \)-shape, \( \alpha \) radius was tuned from 0 to 200 with an interval 2. For each experimental setting, the 100 replications of input points were processed under each discrete parameter value. The parameter value that provided the highest median area-based F1-score of the 100 replications was used as the optimal parameterization for the experimental setting.

### 4.2. Experiment 1: noise to signal ratio

In the first experiment, the accuracy of \( \chi \)-outline, DBSCAN plus \( \chi \)-shape, and modified \( \alpha \)-shape was compared under different noise (outlier) to signal (sample points) ratios (NSR). The number of outliers in the simulation was changed, setting NSR to be 0%, 10%, 20%, 30%, 40%, and 50%.

Figs. 4(a), (c), and (e) show the median area-based F1-scores of the footprints of the letter shapes, border of USA, and border of Malaysia, respectively, constructed using the above three different methods under changes to NSR. All three methods performed better on the country shapes (Figs. 4(c) and (e)) than on the letter shapes (Fig. 4(a)). The increasing number of outliers had more of an effect on letter shapes than on the country shapes. Since the country shapes were reconstructed with a remarkably high accuracy (F1-score over 0.92), the performance difference between the three different footprint methods was extremely small. However, the accuracy of the three methods was clearly different for the letter shapes.

Figs. 4(b), (d) and (f) illustrate the difference of median area-based F1-scores between \( \chi \)-outlines and modified \( \alpha \)-shapes/DBSCAN plus \( \chi \)-shapes for the letter shapes, border of USA, and border of Malaysia, respectively. A positive value means that the \( \chi \)-outline accuracy outperformed that of the alternative method. Conversely, a negative value indicates that the \( \chi \)-outline was less accurate than the alternative method. The nonparametric Kruskal–Wallis test was used to test the null hypothesis that the observed differences between F1-scores for the different treatments might have occurred by chance. The pentagrams above or under the bars indicate the differences are significant at the 5% level (i.e., null hypothesis is rejected in those cases). In general, the graphs show the effect size was greatest for smallest NSR ratio, shrinking with increasing NSR. For the letter shapes, the \( \chi \)-outline accuracy significantly outperformed that of the modified \( \alpha \)-shape at a NSR 30% or less. The accuracy of the \( \chi \)-outline also significantly outperformed the combined DBSCAN plus \( \chi \)-shape at the 0% and 10% NSRs. In all other cases, no significant differences between \( \chi \)-outline and modified \( \alpha \)-shape/DBSCAN plus \( \chi \)-shape were found (i.e., in no cases did modified \( \alpha \)-shape or DBSCAN plus \( \chi \)-shape significantly outperformed \( \chi \)-outlines). For the country shapes, the accuracy differences between \( \chi \)-outlines and modified \( \alpha \)-shapes were positive and significant only at a low NSR (10% or less for USA, 20% or less for Malaysia). Modified \( \alpha \)-shapes significantly outperformed \( \chi \)-outlines only in one case—at the 50% NSR for USA—although the effect size was extremely small. The \( \chi \)-outline significantly outperformed the DBSCAN plus \( \chi \)-shape at all NSRs for Malaysia. For USA, DBSCAN plus...
χ-shape significantly outperformed the χ-outline again with a small effect size at the 40% and 50% NSRs.

As the letter shapes were more difficult (lower F1-score) to reconstruct than the country shapes for all three algorithms, we focused on the letter shapes in the following two experiments.

4.3. Experiment 2: homogeneity

Experiment 1 on the letter shapes demonstrated that χ-outlines performed at least as well as combined DBSCAN plus χ-shapes or modified α-shapes, and tended to be more accurate than these alternatives at lower NSRs. However, as NSR increased, the accuracy advantage of χ-outlines decreased.

To investigate this effect further, the second experiment examined the effect of decreasing homogeneity in point distributions on the performance of the three methods. To achieve this, Experiment 2 fixed the NSR at a relatively high level, 40% (the smallest NSR at which χ-outlines did not significantly outperform the other two alternatives). The experiment then reduced the normalized minimum allowed pairwise distance r between semi-random sample points. The distance r was reduced from $3 \times 10^{-3}$ (the distance used in Experiment 1 and Fig. 4) to 0 (truly random point distributions).

The median area-based F1-scores of the three methods are shown in Fig. 5(a). Inhomogeneous sample points can make some area of the ground truth covered by an insufficient number of points. Hence as expected, the accuracy of all three methods reduced substantially with decreasing homogeneity. Fig. 5(b) illustrates the F1-score differences between χ-outlines and modified α-shapes/DBSCAN plus χ-shapes. The nonparametric Kruskal-Wallis test shows that χ-outlines never significantly outperformed the other two techniques. In most cases

Fig. 4. Effect of different NSRs on the area-based F1-score.
where the normalized minimum allowed distance is $1.5 \times 10^{-3}$ or less,
the $\chi$-outline was outperformed by both the combined DBSCAN plus
$\chi$-shapes or modified $\alpha$-shapes, significant at the 5% level. $\chi$-outlines
were also significantly outperformed by DBSCAN plus $\chi$-shapes for
$r = 2 \times 10^{-3}$ and $r = 2.5 \times 10^{-3}$.

4.4. Experiment 3: non-uniform cluster density

As discussed in Section 3.2, the edge length threshold of $\chi$-outline al-
gorithm was developed to adapt to each connected component in the
triangulation. In contrast, the modified $\alpha$-shape uses a global $\alpha$-radius
to remove triangles from the DT of input points. To demonstrate the ad-
vantage of the $\chi$-outline algorithm in dealing with non-uniform cluster
density, Experiment 1 was repeated, where the input points closest to
different letters were scaled by a different factor. Specifically, as
shown in Fig. 6(a), the input points closest to the letters “C,” “S,” “W,”
“H,” “E,” and “X” were scaled by a factor 0.5, 0.6, 0.7, 0.8, 0.9, and 1.0,
respectively. Fig. 6(b) illustrates the Incremental K-function (Tao, Thill, &
Yamada, 2015) of the points shown in Fig. 6(a). The multiple peaks re-
fl ect that the clusters in the input points have distinctly different densi-
ties. This experimental setup provided a good mixture of different point
densities, which contained clusters of slightly different densities
(e.g., points closest to the letters “E” and “X”) and significantly different
densities (e.g., points closest to the letters “C” and “X”).

As demonstrated by Estivill-Castro and Lee (2002), adaptive cluster-
ing algorithms outperform DBSCAN for discovering clusters of different
densities. But DBSCAN was still used in this experiment to cluster the
input points. This enabled the comparison between the results of this
experiment and those of the first experiment, which can illustrate the
impact of non-uniform cluster density on the two-step clustering-
footprint generation.

The median area-based F1-scores of DBSCAN plus $\chi$-shape, modified
$\alpha$-shape, and $\chi$-outline are illustrated in Fig. 7(a). Fig. 7(b) illustrates
the F1-score differences between $\chi$-outlines and modified $\alpha$-shapes/
DBSCAN plus $\chi$-shapes. In this experiment $\chi$-outlines significantly
outperformed modified $\alpha$-shapes at all NSRs and significantly
outperformed DBSCAN plus $\chi$-shapes at NSRs 0% and 30% and above.
Comparing the results in Fig. 7(a) with the those in Fig. 4(a), the
area-based F1-scores of all three methods were reduced due to
non-uniform cluster density. However, the extents of reduction were
different. Fig. 8 depicts the reduction in area-based F1-scores of the
three methods resulted from non-uniform cluster density. Non-uniform cluster density had the strongest impact on modified
$\alpha$-shapes. DBSCAN plus $\chi$-shapes were affected a little less strongly
than modified $\alpha$-shapes. In addition, the effects of non-uniform cluster
density on modified $\alpha$-shapes and DBSCAN plus $\chi$-shapes were related
to NSR. The area-based F1-scores decreased by a bigger extent when
there were more outliers. The area-based F1-score of modified
$\alpha$-shape reduced by about 5.2% at the 50% NSR. Conversely,
non-uniform cluster density had a much weaker effect on $\chi$-outlines.
The reduction of $\chi$-outlines was irrelevant to NSR. The maximum
reduction of $\chi$-outlines was only 2.1% at the 40% NSR.

Fig. 5. Effect of different homogeneity in point distribution on the area-based F1-score.

(a) Distribution of the input points (b) Incremental K-function of the input points

Fig. 6. Clusters with non-uniform density.
5. Case study

The $\chi$-outline can be applied to a wide range of real-life applications where it is necessary to estimate geographic regions from imprecise point observations of the regions. The experiments above show that the approach is particularly advantageous in cases where cluster and outliers exist, and where point densities in clusters may systematically vary.

Fig. 9 illustrates the use of $\chi$-outlines in a case study of the 2014 Mickleham-Dalrymple wildfires in Victoria, Australia. The gray regions are the true areas impacted by the wildfires, reconstructed by qualified fire mapping officers using a combination of airborne infrared scanner data and human observations. These regions form our ground truth for this case study. However, in practice in the circumstances surrounding an emergency there exist significant practical difficulties in capturing complete and up-to-date extents for rapidly changing wildfires. For many wildfires, such authoritative wildfire extents do not exist, and so there is an interest in using crowd-sourced data to capture this information.

As an alternative to manual expert creation of wildfire extents, the black dots in Fig. 9 represent the actual locations of reports of wildfires during the Mickleham-Dalrymple wildfires, made in calls to the emergency services by the general public. In such a situation, footprint algorithms, like $\chi$-outlines, could be used to estimate automatically the extent of the area impacted by the wildfires, based purely on the locations of wildfire reports from emergency calls. Crucially, in relation to the $\chi$-outline algorithm:

1. Multiple distinct wildfires often occur during the same emergency, necessitating the generation of disconnected polygonal areas automatically distinguishing wildfire reports relating to different disconnected firefronts.
2. Even during an major emergency, instances of irrelevant, nuisance, or incorrect wildfire reports to the emergency services are not uncommon. Hence, any estimation of firefront locations from wildfire reports should be tolerant to outliers.
3. The density of calls relating to different but contemporaneous wildfires can vary systematically by cluster. Wildfires occurring in more remote, less populated areas tend to have a lower density of calls than wildfires occurring in more densely populated areas. The black lines in Fig. 9(a)–(c) are the optimal footprints of the wildfire extents derived from emergency call reports and generated using DBSCAN plus $\gamma$-shape, modified $\alpha$-shape, and $\chi$-outline, respectively. Again, all the optimal footprints matched the ground truth relatively accurately. However, the area-based F1-scores shown in Fig. 10(a) illustrate that the footprint generated using the $\chi$-outline algorithm was the most accurate, achieving a remarkably high area-based F1-score of 82.8%. The $\alpha$-shape was the least accurate method, with an area-based F1-score of 75.4%.

The area-based F1-score tends to be biased towards large ground truth objects and large estimation polygons. The reliance on area makes the metric insensitive to the errors of small false positive estimations and small false negative ground truth objects. However, from the perspective of emergency management, such errors of commission and omission of small fires may have significant impacts, such as the potential for incorrectly diverting limited firefighting resources to apparently “new” fire locations. Hence, the average per-object area-based recall and the average per-estimation area-based precision (Nascimento & Marques, 2006) of our estimates were also calculated. The average per-object area-based recall is calculated as the mean of the area-based recall for all ground truth objects. Similarly, the average per-estimation area-based precision is computed as the mean of the area-based precision for all estimated polygons. The average area-based F1-score is then simply the harmonic mean of average per-object area-based recall and average per-estimation area-based precision. This metric better accounts for errors of commission of all estimates, and errors of omission of ground truth objects. The results shown in Fig. 10(b) indicate substantial increases in accuracy for the $\chi$-outlines, when compared with DBSCAN plus $\gamma$-shape and the modified $\alpha$-shape. These results arose because the $\chi$-outline generated fewer false positive estimation polygons than
the DBSCAN plus χ-shape; and it estimated the extents of fires more accurately than the modified α-shape.

6. Discussion and future work

In the categorization of footprint algorithms proposed by Galton and Duckham (2006), for the χ-outline: outliers are permitted; output polygons can comprise multiple disconnected components, if input points contain multiple clusters; and the polygonal shapes constructed by the χ-outline algorithm can have point connections.

The optimal χ value for the χ-outline algorithm in the experiments was found to decrease with increasing NSR. The optimal χ value reduced significantly from 1.0 to 0.6 when NSR was changed from 0% to 10%. For NSRs higher than 10% (20%–50%), the optimal χ value decreased gently from 0.4 to 0.3. Hence if the input data set does not contain any noise, a high χ value (e.g., 1.0) is expected to provide near-optimal results. A moderate χ value (e.g., 0.6) is more likely to handle slightly noisy input data well. A small χ value (e.g., 0.4) is more appropriate when the input data is anticipated to contain a large proportion of noise points.

The empirical experiments and the case study demonstrate that optimal performance of the χ-outline algorithm can equal and even exceed the optimal performance of the two-step clustering-footprint technique. However, using a clustering algorithm like DBSCAN introduces two additional parameters, with any footprint algorithm requiring at least one further parameter. Although the combination of clustering and footprint has the advantage of modularity, reusing common existing tools, its disadvantage is in the additional parameters required. The combination of three parameters makes achieving optimal performance in practice even harder. By contrast, the χ-outline algorithm does not need pre-clustering, has one single parameter, and is therefore arguably easier to be adjusted for desirable results than a two-step clustering-footprint approach.

The χ-outline algorithm can even outperform the two-step clustering-footprint technique when the cluster density is non-uniform. This is because DBSCAN was not developed to handle clusters with different densities. In this situation, adaptive clustering algorithms such as AUTOCLUST (Estivill-Castro & Lee, 2002), ASCDT (Deng, Liu, Cheng, & Shi, 2011), and NSCABD (Yang & Cui, 2008) can distinguish clusters and outliers more accurately than DBSCAN. However, these adaptive clustering algorithms are not better than DBSCAN in all cases. Thus, introducing a range of more specialized clustering algorithms does not guarantee improved performance, and further complicates the task of footprint generation for the user: not only does the parameterization of the clustering algorithm need to be adjusted, but also the selection of the clustering algorithm may need to be changed from case to case. The χ-outline algorithm uses an adaptive length threshold to split the DT of input points into multiple connected components and to construct the shapes of the connected components. Hence it can handle complicated distributions in input points.

Fig. 9. Case study: estimating boundary of 2014 Mickleham–Dalrymple wildfires in Victoria, Australia.

Fig. 10. Area-based F1-score (a) and average area-based F1-score (b) of the affected area of Mickleham–Dalrymple wildfires estimated by DBSCAN plus χ-shape, modified α-shape, and χ-outline.
χ-outlines always remove exterior edges to construct the footprint. Hence χ-outlines are always regular polygonal shapes. In contrast, α-shapes may need to be regularized to obtain regular polygonal shapes. In addition, exterior edges are removed in order, therefore the χ-outline algorithm prioritizes the separation of clusters and removal of outliers. Then the χ-outline algorithm assigns adaptive length thresholds to different connected components using a simple but robust heuristic. The heuristic takes into account both local and global statistics of the edge lengths of the connected components. So χ-outlines are more sensitive and adaptive than α-shapes to the characteristics of the boundary of the distribution of input points. As a result, χ-outlines outperform α-shapes when the distribution of input points is homogeneous (e.g., in our first experiment).

The basic α-shape performs the worst when input points consist of clusters with non-uniform point densities (e.g., in our third experiment and case study). Extended algorithms such as weighted α-shape (Edelsbrunner, 1992), conformal α-shape (Cazals, Giesen, Pauly, & Zomorodian, 2005), and LDA-α-shape (Maillot, Adam, & Melkemi, 2010) were developed to handle complicated distributions in input points. The χ-outline algorithm explores a new approach to handling complicated distributions in input points. The ordered edge removal enables an adaptive technique to separate disconnected shapes and trim the boundary of the shapes. Thus, the χ-outline algorithm is a useful supplement to existing footprint construction algorithms.

The second experiment (homogeneity) revealed a disadvantage of χ-outlines: being outperformed by both the combined DBSCAN plus χ-shapes or modified α-shapes when the distribution of input points was strongly inhomogeneous. This is because strong inhomogeneity in the point distribution can cause extremely uneven edge lengths in the intermediate connected components of the χ-outline algorithm; and the length thresholds of exterior edges are adaptive at the connected component level. As a DT based algorithm, it is easy to extend the χ-outline algorithm further to adapt the length threshold of an exterior edge to the local features of the edge, like the local features used in AUTOCLUST, ASCDT, and NSCABD. It is also possible to use the “neck” identification and removal schemes proposed in AUTOCLUST and ASCDT to assist the separation of clusters. These extensions to the current χ-outline algorithm may address the above disadvantage of χ-outlines. Our future work will focus on the development and test of the more adaptive version of the χ-outline algorithm.

7. Conclusion

A new footprint χ-outline is proposed in this paper. Based on a modest extension of the established χ-shape algorithm, our χ-outline algorithm handles points which have a spatially clustered pattern and outliers. Experimental results, based on both simulated data and real crowd-sourced data connected with wildfire extents, demonstrate that χ-outlines can reconstruct the target regions as least as accurately as combined DBSCAN plus χ-shapes or as α-shapes. In many cases, χ-outlines can improve upon these state-of-the-art alternatives, especially when the density of input point clusters varies systematically. This feature is not uncommon in real applications, such as our case study of emergency wildfire reports, where the density of calls clusters tends to vary based on the underlying population density.

χ-outlines achieve these results without the need to pre-clustering input points, which reduces the complexity and potential for inappropriate parameterization of clustering algorithms. The output of the χ-outline algorithm consists of regular polygonal shapes. The χ-outline algorithm is as efficient as α-shapes or χ-shapes, O(n log n); flexible and can be controlled by a single parameter; and can generate the perceived shapes for the spatial pattern in the distribution of input points.

Acknowledgments

This work is funded by the ARC Linkage Project (Grant No. LP120020584) “Resilient Information Systems for Emergency Response (RISER).”

References