Optimization of 2D Surface Detection and Edge Reconstruction using Alpha Shape

Computer and Systems Engineering Department, Alexandria University
MSc Pattern Recogntion Course Project
Supervised by: Prof. Dr. Mohamed Ismail
Prepared by: Dina Samak

***Abstract*—In this work we are introducing a method to optimally select the value of alpha used to determine the optimal or at least sub-optimal alpha shape used to detect the surface of a 2D polygon represented by a set of multiple points in the 2D space and reconstruct its edges.**

I. Introduction

The description of shapes is considered a vital problem in the computer graphics field. 2D shapes representation is an essential topic that is used in many fields such as:

- Determining places and sites on maps.
- Describing the location of a military convoy consisting of a set of vehicles
- Defining the boundaries between neighborhoods of a city consisting of sets of buildings.
- Diagnosis of malfunctioned part in medical images.
- Completing letters for non-obvious handwriting.

In general it is used to describe 2D spatial clusters in Machine Learning Clustering Problems or Community Detection in graphs.

Polygons are used to describe the area of the given 2D cluster and the edges of this polygon are the contour of this cluster. Thus the choice of the exact polygon that represents the cluster is an important task such that it should include all points of the cluster as well as it should not contain empty areas that results of non-exact representation of the cluster or unneeded overlapping of neighboring clusters that may output non-accurate results. Fig.1 shows the exact and non-exact representations of set of points in a cluster.

II. Related Work

Convex hulls are the simplest way to enclose a set of points in a polygon. However, convex hulls may contain large empty areas that are not desirable. Creating polygon models based on Voronoi diagrams or Delaunay triangulations is another commonly used approach. Matt Duckham et al. [8] propose a "simple, flexible, and efficient algorithm for constructing a possibly non-convex, simple polygon that characterizes the shape of a set of input points in the plane, termed a Characteristic shape". The algorithm firstly creates the Delaunay triangulation of the point set which actually is the convex hull

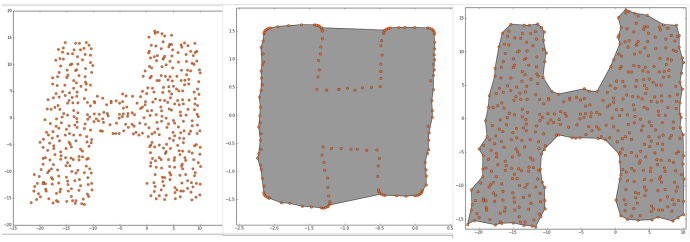

Fig. 1. Cluster points with the non-exact representation in the middle exact representation on the right.

of the point set and then reduces it to a non-convex hull by replacing the longest outside edges of the current polygons by inner edges of the Delaunay triangulation until a termination condition is met. The Alpha shapes algorithm, introduced by Edelsbrunner et al. [3] also uses Delaunay triangulation as the starting step and generates a hull of polylines, enclosing the point set and this hull is not necessarily a closed polygon. Thus, the Alpha shapes algorithm requires post-processing for creating polygons out of the polylines. Besides, there is no easy way of determining the proper parameter for Alpha shapes algorithm.

III. Alpha Shape Optimization

A. Main Algorithm

The α-shape of the given set of points is the simplicial complex formed by the set of edges and triangles whose radii are at most $1/\alpha$.

To Compute alpha shape:

Algorithm 1: Get Alpha Shape

```
Result: Edges of the Polygon
Get the Delaunay triangulations of the set of Points;
for each triangle in the triangulations do
    compute area and parameter;
    if area >0 and a*b*c/(4*area)< 1/α then
        Add all the triangle sides into the alpha shape
         list;
    else
        Get the union of all the triangles formed;
    end
end
```

B. Alpha Value Optimization

Getting the optimal value for alpha is our main concern. Large values of alpha results in an empty cluster with no points, relatively large values of alpha make the cluster misses far points and does not contain all of the cluster points, relatively small values gives non-exact representation and zero gives convex hull.

In April 2019, an open source Alpha Shape Tool Box [23] [24] was implemented under an MIT license to draw alpha-shapes using Shapely polygons in Python. This toolbox included a function used to optimize alpha. It used Binary Search technique to find optimal Alpha Value, with default maximum iterations equal to 10000.

We introduce a less processing time implementation that gives the same result, yet it reduces the computation time dramatically. By assign a pre-computed array of numbers that have numbers that range from Large predetermined number to zero, with varing steps (large difference in large numbers and small difference in small numbers). For example: alpha arr = $[5000, 3000, 1000, 500, 300, 100, 50, 25, 18, 10, 5, 0]$

Whenever a sub-optimal number arises, we take the range between that number and the number before in the array, and apply binary search on, that gives the optimal alpha. We added the assumption that alpha is a positive integer (or zero in the convex hull case) which didn't change in the results.
We took in account the constraint of including all cluster points in the resultant polygon and excluding the cases where multi-polygons arises (not complete polygons with empty space in the middle).

IV. Case Study - Clusters Representation

A. Dataset

UK police forces collect data on every vehicle collision in the uk on a form called Stats19. Data from this form ends up at the DfT and is published at https://data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data.
Also found on Kaggle: https://www.kaggle.com/silicon99/dft-accident-data Used KPIs: Latitude, Longitude and Local Authority (Highway) Where the coordinates shows the place of the accident in UK, and the local authority ID is the region which is had for the accident in. We used the uk-accidents dataset of year 2014, it has 207 regions with total number of accidents equal to 146322 accidents (number of samples).
Fig. 2 shows the polygon clusters when computed using the optimized technique.

B. Comparative Study

However we could conclude the output from our algorithm implementation, the other algorithm we are comparing our results to will take a lot of time (more than 24 hours) to give out similar results. So for the sake of comparison we shall choose 33 regions in the North of UK to set our comparison of total 207 regions. Number of samples chosen is 8,790 from total 146,322 samples . Fig.3 shows that convex hull representation will not be reliable nor accepted in most of the cases. While the optimized alpha shape and the original implementation merely shows any difference. Fig 4 shows a boxplot representing the difference between alpha values in all polygons.

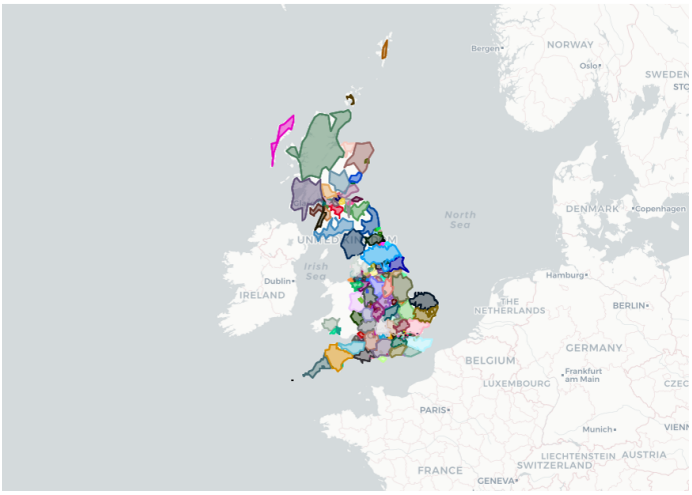

Fig. 2. UK accidents Clusters Using Optimized Alpha Shape

TABLE I
Processing Time for each Algorithm

Original max iteration =2000	Optimized Alpha	Convex hull (alpha = 0)
5529.875 sec.	0.48 sec.	0.7 sec.

The great distinguish between the 2 algorithms (the original one and the introduced one) is in the processing time. The original took more than 12 hours with 10000 iterations and with only 2000 iterations took an hour and half, the optimized took 48 seconds, while the convex hull only took 0.8 sec. This great distinguish in the processing time arises from the number of iterations used to reach optimum state. The original algorithms ranges from the maximum available value to zero and performs binary search over all this range. Meanwhile, the introduced algorithm only iterates between small range after performing a step of choosing a sub-optimal alpha value.
Moreover, another reason for this distinguished difference is the using of integers in the optimized model rather than iterating over infinite floats that in most cases makes the algorithms stops when reaching maximum iteration value. Also it is worth mentioning that decreasing the max iteration value does not give optimum results.

Another approach is conducted in this study, it's the Concave hull algorithm using a K-nearest neighbors approach [12]. This Algorithm shows a different representation for the 2D surface, yet we can't tell which algorithm best describe the surface optimally. The only algorithm's draw back that it takes a very long time in detecting the surface of of a cluster that have a point that is far from other points of the clusters. The concave hull algorithm took about 4 minutes to detect the surface of the 33 clusters, with excluding the region that have a point with a large distance from other dense points.

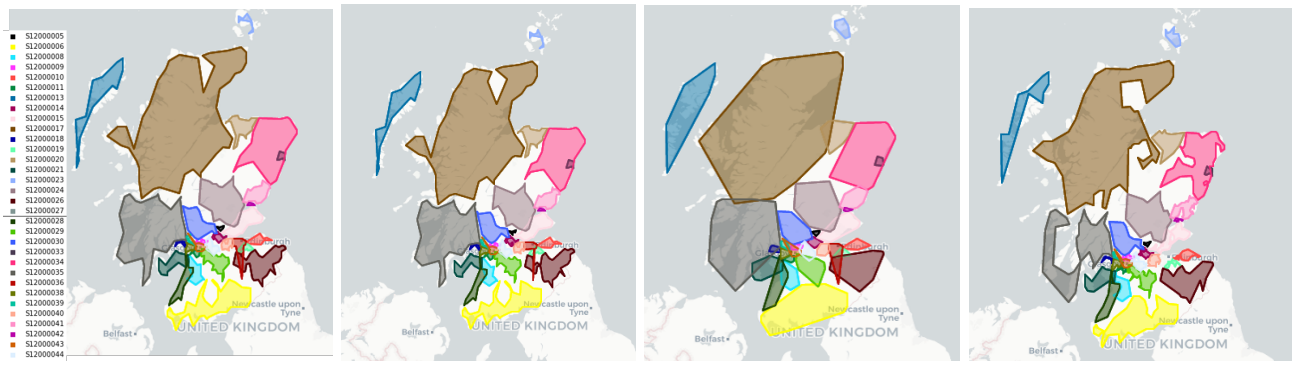

Fig. 3. Original alpha value algorithm results VS Optimized alpha Value Results VS Convex hull results VS Concave hull

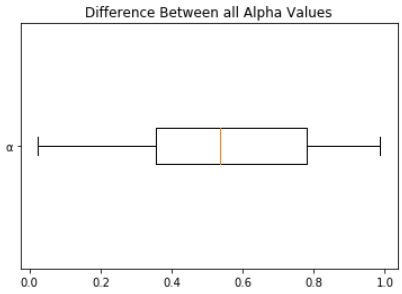

Fig. 4. Box Plot of The difference between the chosen alpha with and without optimization.

V. Conclusion

Polygons are the simplest way to represent spatial 2D clusters. And the problem to determine the this polygon and reconstruct its edges is a vital problem that is used in multiple aspects in industry, maps, words, images,etc. Convex hull is a quick solution for spatial clusters that do not need exact boundary representation.

Yet, whenever exact cluster representation is essential, the optimized approach would give similar results with dramatic improvement of processing time.

References

[1] Akdag, Fatih and Eick, Christoph and Chen, Guoning. (2014). Creating Polygon Models for Spatial Clusters. 8502.

[2] C. Mineo, S.G. Pierce, R. Summan, Novel algorithms for 3D surface point cloud boundary detection and edge reconstruction, Journal of Computational Design and Engineering (2018), doi: https://doi.org/10.1016/j.jcde.2018.02.001.

[3] Edelsbrunner, H., Kirkpatrick, D.G., Seidel, R.: On the shape of a set of points in the plane. IEEE Transactions on Information Theory 29, 551–559 (1983).

[4] Dmitry Krasnoshchekov, Valentin Polishchuk,Order-k -hulls and α-shapes, Information Processing Letters, Volume 114, Issues 1–2, 2014, Pages 76-83, SSN 0020-0190, https://doi.org/10.1016/j.ipl.2013.07.023.

[5] Christos Varytimidis, Konstantinos Rapantzikos, Yannis Avrithis, Stefanos Kollias, α-shapes for local feature detection, Pattern Recognition, Volume 50, 2016, Pages 56-73, ISSN 0031-3203, https://doi.org/10.1016/j.patcog.2015.08.016.

[6] B. Delaunay, "Sur la sphere vide. A la memoire de Georges Voronoi", Bulletin de l'Academie des Science URSS. Classe des sciences mathematiques et na, 1934, no. 6, 793–80.

[7] SONG Zhan feng, PU Hao, ZHAN Zhen yan (School of Civil and Architectural Engineering. Central South University, Changsha 410075, China);Study on an algorithm for fast constructing Delaunay triangulation[J];Journal of The China Railway Society;2001-05.

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

[9] C. Varytimidis, et al., α-shapes for local feature detection, Pattern Recognition (2015), http://dx.doi.org/ 10.1016/j.patcog.2015.08.016i

[10] F. Cazals, J. Giesen, M. Pauly, A. Zomorodian, Conformal alpha shapes, In: Eurographics/IEEE VGTC Symposium Proceedings of Point-Based Graphics, IEEE, New York, USA, 2005, pp. 55–61.

[11] H. Edelsbrunner, Weighted Alpha Shapes, University of Illinois at UrbanaChampaign, Department of Computer Science, 1992.

[12] Adriano Moreira and Maribel Yasmina Santos (2007): CONCAVE HULL: A K-NEAREST NEIGHBOURS APPROACH FOR THE COMPUTATION OF THE REGION OCCUPIED BY A SET OF POINTS. GRAPP 2007 - International Conference on Computer Graphics Theory and Applications, pp. 61-68

[13] Fadili, M. J., Melkemi, M., ElMoataz, A. (2004). Non-convex onion-peeling using a shape hull algorithm. Pattern Recognition Letters, 25(14), 1577–1585.

[14] J.-F. Dufourd and F. Puitg, "Functional specification and prototyping with oriented combinatorial maps," Computation Geometry—Theory and Applications, vol. 16, no. 2, pp. 129–156, 2000.

[15] Park, J.-S Oh, S.-J. (2013). A New Concave Hull Algorithm and Concaveness Measure for n-dimensional Datasets. Journal of Information Science and Engineering. 29. 379-392.

[16] Fortune, S. Stable maintenance of point set triangulations in two dimensions. Proceedings of the 30th annual IEEE Symposium on Foundations of Computer Science Vol. 30, 494-499, 1989.

[17] Kettner, Lutz; Mehlhorn, Kurt; Pion, Sylvain; Schirra, Stefan; Yap, Chee (2008). "Classroom examples of robustness problems in geometric computations" (PDF). Computational Geometry. 40 (1): 61–78

[18] C. Harris, M. Stephens, A combined corner and edge detector, In: Alvey Vision Conference, 1988, pp. 147–151.

[19] K. Rapantzikos, Y. Avrithis, S. Kollias, Detecting regions from single scale edges, In: International Workshop on Sign, Gesture and Activity (SGA), European Conference on Computer Vision (ECCV), Heraklion, Greece, Lecture Notes in Computer Science, vol. 6553, Springer Berlin Heidelberg, 2010, pp. 298–311.

[20] H. Meine, U. Köthe, P. Stelldinger, A topological sampling theorem for robust boundary reconstruction and image segmentation, Discrete Appl. Math. 157 (3) (2009) 524–541.

[21] H. J. Miller and J. Han, Eds., Geographic Data Mining and Knowledge Discovery. CRC Press, 2001.

[22] J.F.Canny,A computational approach to edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 8(6)(1986)679–698.

[23] https://pypi.org/project/alphashape/.

[24] https://alphashape.readthedocs.io/en/latest/.

[25] https://github.com/joaofig/uk-accidents.

[26] http://blog.thehumangeo.com/2014/05/12/drawing-boundaries-in-python/