World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:5, No:5, 2011

International Science Index, Computer and Information Engineering Vol:5, No:5, 2011 waset.org/Publication/5368

Constructing a Simple Polygonalizations

V. Tereshchenko and V. Muravitskiy

Abstract—We consider the methods of construction simple polygons for a set *S* of *n* points and applying them for searching the minimal area polygon. In this paper we propose the approximate algorithm, which generates the simple polygonalizations of a fixed set of points and finds the minimal area polygon, in $O(n^3)$ time and using $O(n^2)$ memory.

Keywords—simple polygon, approximate algorithm, minimal area polygon, polygonalizations

I. Introduction

The problem of generating random geometrical figures, except existing theoretical interest, is motivated by the need to generate test data to verify the correctness and time complexity of algorithms of computational geometry [1]. There is a wide range of possible applications for the algorithms, associated with the construction of simple minimal area polygons, in GIS systems, [2, 3]. Another direction of application in this area is geo-sensor networks [4, 5].

One of the problems is the impossibility of calculating the number of simple polygons for a given set of points, in polynomial time. An important problem is finding a simple polygon that have certain properties. In particular, search of the minimal area polygon among all possible polygons which can be generated on a given set of points. The problem of finding a simple polygon of minimal area has got certain weight in pattern recognition problems. Therefore the search of the optimal algorithm that can generate a simple polygon of minimal area is still actual.

Analysis of recent research. For today, there are several approaches to the solution of the problem, that based on using Delaunay triangulation or Voronoi diagram. In [6, 7] the authors introduce "α-shape" -notion for the generalization of a convex hull that allows to develop methods of constructing the simple polygons with using Voronoi diagram. In [8-10], "A-shape" used to "onion-peeling"- method, by removing the boundary edges of triangulation [11]. The papers [12,13] propose algorithms using Delaunay Triangulation. The approach allows us to develop a simple, flexible and efficient algorithm for constructing a simple polygon using the notion of characteristic form. The characteristic form is varied from a convex hull to the uniquely defined form of a minimum area. It experimentally confirms that the algorithm, using an appropriate parameterization, can precisely construct the characteristic forms for different sets of points. Another algorithm was proposed in [14]. The algorithm starts with constructing a convex hull, and then uses the procedure of "divide-and-conquer, successively inserting additional edges and smoothing zigzags. The complexity of the algorithm is limited by complexity of constructing the initial convex hull - $O(n \log n)$.

Moreover, we can distinguish two approaches to the generation of simple polygons for a given set of points. The first suggests that we need to find a single polygon without consideration of its properties [1]. The second approach provides algorithms that generate "random" polygons for this set. This problem is more complicated, but efficient algorithms were proposed to solve it [1, 15]. Attempts to solve the problem of finding a polygon of minimal area also have been undertaken, and have achieved certain results for its solution [16].

The novelty and idea. In the paper we propose a polynomial approximation algorithm for the minimal area polygon.

Paper's aim. Explore algorithms for generation of simple polygons given a set of vertices, and develop an algorithm for determining the minimal possible area polygon.

II. Problem and algorithms

Problem. *Let S – given set of n points on the plane. 1. It is necessary to specify the order of connection by edges of points from set S so to generate simple polygons. 2. Find among the generated polygons the minimum area polygon.*

In solving the given problem, we can distinguish algorithms that build a simple polygon (unique requests), algorithms that generate all possible set of simple polygons (mass requests) and algorithms for finding the minimal area polygon.

A. Constructing a simple polygon for a given set of points (unique requests)

In this case, we can suggest the following algorithms.

Algorithm 1. Choose $P_0 \in S$ as anchor point. It is the start of bypass. Sort all other points $\{ S \setminus P_0\}$ with polar angle relative P_0. As a result, we get one of the possible polygons for a given set *S*. Sorting can be done in $O(n \log n)$ time

Algorithm 2. Constructing convex hull for *S*. If all points of *S* belong to a contour, then problem is solved. If not, look for a point P_1, which is at minimum distance from the contour – even this minimum distance to the side pieces (P_{k-1}, P_k). If we have several such points, we take any of them. Insert point

V. Tereshchenko, Taras Shevchenko National University of Kyiv, Kyiv, Ukraina ((phone: +3044-259-04-27; fax: +3044-259-04-39; e-mail: vtereshch@gmail.com) .

V. Muravitskiy, Taras Shevchenko National University of Kyiv, Kyiv, Ukraina ((phone: +3044-259-04-27; fax: +3044-259-04-39; e-mail: vmyrik@gmail.com) .

International Science Index, Computer and Information Engineering Vol:5, No:5, 2011 waset.org/Publication/5368

P_1 in the contour that is instead of (P_{k-1}, P_k) will be (P_{k-1}, P_1, P_k). For all the points that remain, we repeat the above procedure until the last point will be inserted in the contour.

Algorithm 3. Building convex hull for *S*. If all points of *S* belong to a contour, then problem is solved. Otherwise, denote all the internal point S_1 of a convex hull. Building for a new set S_1 convex hull V_1(contours *V* and V_1 do not intersect). Sticking together two contours. Choose a pair of consecutive vertices *u*, *v* and u_1, v_1 on contours *V* respectively, so that in the quadrangle with vertices *u, v,* v_1, u_1 not lay any more points from contours. *V* and V_1. Severing contours *V* and V_1 (removing edges (*u*, *v*) and (u_1, v_1)) and connecting them (adding edges (u, u_1) and (v, v_1)).

If V_1 does not contains the points, the problem is solved. Otherwise, we conduct the same operations with internal points: we find the convex hull and a pair of consecutive points on the contours; we couple and uncouple contours until we get a convex hull which includes 0, 1 or 2 points. If count of points equals 0, then problem is solved. Otherwise, we add points to the contour so that the figure remained polygon (can be conducted joining, as in previous case).

B. Generating the simple polygons for a given set of points (mass requests)

In this case we can use the following algorithm [15].

Steady Growth . At initialization, **Steady Growth** randomly selects 3 points s_1, s_2, $s_3 \in S$ such, that no other points no lie outside CH({s_1, s_2, s_3 }). Let $S_1 = S \setminus \{s_1, s_2, s_3\}$. During the *i*-iterations ($1 \le i \le n$-3) we perform the following operations:

1. We choose randomly $s_1 \in S_i$, but no points $S_{i+1} = S_i \setminus \{s_i\}$ that lie outside CH($P_{i-1} \cup \{s_i\}$) .
2. Finding an edge (v_k, v_{k-1}) in P_i that completely visible of s_i and replacing it by edges (v_k, s_i) and (s_i, v_{k+1}) (s_i , v_{k+1}) .

Permute & Reject. This algorithm works as follows: generates one possible permutations of the set S and involves checking whether is this permutation a simple polygon. If yes - we got the result, otherwise generate another permutation. This algorithm is inefficient when all (or most) points lie on the convex hull, because only 2*n* of *n*! permutations corresponding to simple polygons. Permute & Reject can generate all possible polygons.

So if we need to generate a simple polygon for a given set of points, we can do this by using the above algorithms.

C. Generating the minimum area polygon

The problem of finding the minimum polygon area known as MAP - Minimum Area Polygon. It is NP-complete, which was proved in [16]. That is, for a minimum polygon area, we must review all the possible polygons for a given set *S* of *n* points.

Consider now the same problem with the position of the minimum polygon approximation. Since MAP - NP-complete problem, we can't find an exact polynomial algorithm for finding the minimal area polygon, so we have to find a good approximation method. The main reason lies in the complexity of approximation of complex geometric relationship between the boundary simple polygon and its area. So can be trying some heuristics to try and get a minimum polygon area. One of effective approaches to solve a problem based on the idea of greedy algorithm "Greedy - build" [16]. We start with the smallest not degenerate triangle in the set *S*. While the vertex is not included in the polygon, we choose the smallest not degenerate triangle that formed the current polygon edge and vertex outside polygons that completely "sees" edge. We adding to the triangle polygon and back to previous step.

We propose the following algorithm for finding the minimum area simple polygon.

III. The Proposed (Greedy) Approximation Algorithm for Minimum Area Polygon

It is invite the following approximation algorithm:

1.Jarvis method for building the convex hull of the set *S* (Figure 1) (this method is best approximations, because if the basic number of points not lying on the convex hull, we'll get it in time *O (hn)*, where *h* – points that lie on the convex hull).

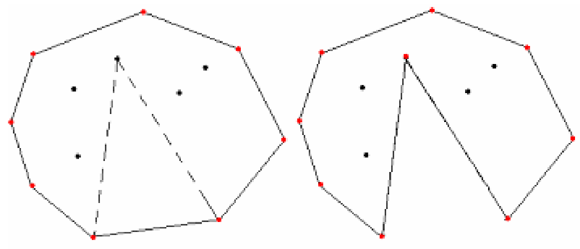

Fig. 1 Point selection and inserting point into the contour

2. While the vertex is not included in the polygon: choose among the points that are inside this polygon point *P*, which will form with one of the edges *(u, v)* the triangle of maximum area, and one that does not contain other points, and also does not intersects the edges of this polygon (the edge "seen" from the point).

3. Let *(u, v)* - edge, *P* - points that formed a triangle of maximum area that satisfied the specified conditions. Severing edge *(u, v)* and form two new *(u, P)* and *(P, v)*. Return to p.2 .

For the proposed greedy algorithm, we obtain the following estimates of complexity. Building convex hull *O (hn)*, in the worst case $O(n^2)$, but this case is both the best, because once we get the solution of the original problem. We need to insert *n* points to the polygon. To insert each of them we take over all edges of this polygon, and for each edge, we take all remaining points. For each point in question we check whether the triangle contains no other points, and whether it crosses no the edges of this polygon. This gives us a total complexity of $O(n^4)$, Using memory - *O (n).*

International Science Index, Computer and Information Engineering Vol:5, No:5, 2011 waset.org/Publication/5368

Optimization of the proposed algorithm. It is possible conduct preliminary processing of the set *S*, which will improve time characteristics of the complexity of this algorithm, but slightly increase memory usage. There is an algorithm in [17], which provides preparation for a set S $O(n^2)$, using the $O(n^2)$, memory that will check whether a triangle contains a set of other points in constant time. The total complexity of this greedy algorithm will be $O(n^3)$. The algorithm builds a matrix stripe $[p_i, p_j]$, elements of which - the number of points that are in the vertical segment below p_i, p_j.

<u>*Algorithm.*</u>

1) Fill all elements stripe [*,*] zeros.

2) Sort out all points by the *x*-coordinate from left to right. This gives the sequence $p_1 \ldots p_n$

3) For each point p_i, sort all the points that lying left of p_i in the clockwise order around p_i. This will give sequences $p_1^i \ldots p_{i-1}^i$

4) For $p_i := p_2$ to p_n:

For $j := 2$ to i-1:

1) if p_j^i left lying on p_{j-1}^i

Stripe$[p_j^i, p_i]$ = stripe$[p_{j-1}^i, p_i]$+stripe$[p_j^i, p_{j-1}^i]$ + 1

2) if p_j^i right lying on p_{j-1}^i

Stripe$[p_j^i, p_i]$ = stripe$[p_{j-1}^i, p_i]$-stripe$[p_j^i, p_{j-1}^i]$

During the study the approximation algorithms, it was found that it is impossible to find a method that would approximate polygon minimum area for a given set of points with constant accuracy. Therefore, there is a question on existence of such method in general. It is possible prove that the approximation of the polygon with constant accuracy it is NP-complete problem. To prove we'll use the principle of reduction problems [18].

As a prototype, we will use Minimum Area Triangulation (MAT) problem, for which we have a proof of NP-completeness.

Minimum Area Triangulation problem (MAT). *On a given set P of 3n points on a plane find a set of disjoint triangles* T_i, *i=1..n, such that the total area* $\sum_i AR(T_i)$ *is smallest possible*.

Lemma. *Minimum Area Triangulation(MAT) problem is reducible to the problem of construction simple polygons in a linear time.*

Proof. The set *S* of *n* points in the plane is an input data for the approximation problem. The same set is an input data for MAT. So input data of approximation problem can be transformed into input data of MAT in the time of *O(1).* We have a proof of NP-completeness of MAT [18, 19].

Output data of approximation problem can be transformed into output data of MAT in linear time. We can use some of triangulation algorithms for it. Hence we have that MAT problem is reducible to the approximation problem. So the approximation problem has the same estimation of complexity as MAT and the polynomial algorithm for solving this problem does not exist.

IV. Implementation

To test the efficiency of the proposed method we made implementation on Java.

To obtain reliable data on the algorithm was necessary to compare results with other algorithms. For comparison was elected algorithm of exhaustive search which is based on the above Permute & Reject: all possible permutations are generated and checked for simplicity. Although the problem is NP-complete for sets of small capacity, it gives the result. Moreover, this result is best possible. (Figure 2).

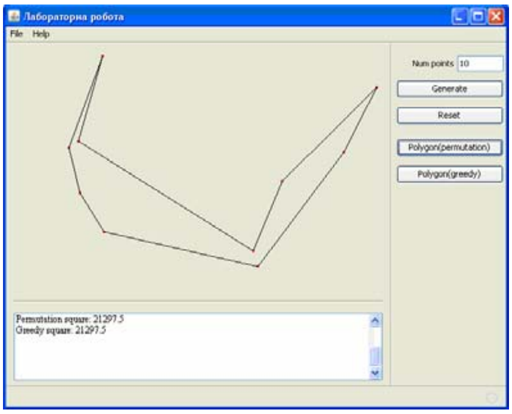

Fig. 2 The result of the program

Figure 2 shows that for the given test set with 10 points, minimal area polygon, which constructed by algorithm of exhaustive search and which generated by greedy algorithm, have coincided.

V. Conclusion

The paper had investigated the problem of generation of simple polygons for a given set S of N points. It was revealed that this problem can have different interpretations, namely one-time generating of a simple polygon or generating some set of simple polygons - random generation. There are efficient algorithms for solving the first problem. They give the result in a polynomial time. There are algorithms for the second case too, but their efficiency is lower. Examples of such algorithms were presented.

Concerning the problem of finding the minimal area polygon, there were some difficulties. Initially it was found that the problem of counting the number of simple polygons for a given set of points is NP-complete, so simple algorithm of exhaustive search is ineffective even if the set S has a small capacity. So researches were switched to approximation methods. It was found that the proposed greedy algorithm with a time complexity O (n^4) and memory usage O (n) can be improved, and we will get the time complexity O(n^3) and memory usage O(n^2). But in practice, the methods gave results that differed from the optimum. Further investigation showed that constant-factor approximation is impossible and it was proved.

REFERENCES

[1] Chong Zhu, Gopalakrishnan Sundaramb, J. Snoeyink and Joseph S. B. Mitchell. Generating random polygons with given vertices. *Jur.Computational Geometry: Theory and Applications* . Vol. 6, Issue 5, Elsevier, 1996, Pages 277-290.

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

[3]] A. Galton and M. Duckham, “What is the region occupied by a set of points?” in GIScience, ser. Lecture Notes in Computer Science. Springer, 2006, vol. 4197, pp. 81–98.

[4] A. Galton, “Dynamic collectives and their collective dynamics,” in COSIT, ser. Lecture Notes in Computer Science. Springer, 2005, vol. 3693, pp. 300–315.

[5] M. F. Worboys and M. Duckham, “Monitoring qualitative spatiotemporal change for geosensor networks,” International Journal of Geographic Information Science, vol. 20, no. 10, pp. 1087–1108, 2006.

[6] H. Edelsbrunner, D. G. Kirkpatrick, and R. Seidel, “On the shape of a set of points in the plane,” IEEE Transactions on Information Theory, vol. IT–29, no. 4, pp. 551–558, 1983.

[7] M. Melkemi and M. Djebali, “Computing the shape of a planar points set,” Pattern Recognition, vol. 33, pp. 1423–1436, 2000.

[8] M. J. Fadili, M. Melkemi, and A. ElMoataz, “Non-convex onion-peeling using a shape hull algorithm,” Pattern Recognition Letters, vol. 25, pp. 1577–1585, 2004.

[9] A. R. Chaudhuri, B. B. Chaudhuri, and S. K. Parui, “A novel approach to computation of the shape of a dot pattern and extraction of its perceptual border,” Computer Vision and Image Understanding, vol. 68, no. 3, pp. 257–275, 1997.

[10] Nina Amenta, , Sunghee Choi and Ravi Krishna Kolluri. “The power crust, unions of balls, and the medial axis transform, jur.*Computational Geometry: Theory and Applications,* vol. 19, no. 2–3, pp. 127–153, 2001.

[11] B. Chazelle, “On the convex layers of a planar set,” IEEE Transactions on Information Theory, vol. 31, pp. 509–517, 1985.

[12] H. Alani, C. B. Jones, and D. Tudhope, “Voronoi-based region approximation for geographical information retrieval with gazetteers,” International Journal of Geographical Information Science, vol. 15, no. 4, pp. 287–306, 2001.

[13] A. Arampatzis, M. van Kreveld, I. Reinbacher, C. B. Jones, S. Vaid, P. Clough, H. Joho, and M. Sanderson, “Web-based delineation of imprecise regions,” Computers, Environment, and Urban Systems, vol. 30, no. 4, pp. 436–459, 2006.

[14] G. Garai and B. B. Chaudhuri, “A split and merge procedure for polygonal border detection of dot pattern,” Image and Vision Computing, vol. 17, pp. 75–82, 1999.

[15] Thomas Auer and Martin Held.Heuristics for the generation of random polygons. In *Proc. of the 8th Canadian Conference on Compu. Geom. (CCCG'96)*, pages 38-43, 1996.

[16] S.P. Fekete, W.R. Pulleyblank. Area Optimization of Simple Polygons, Proceedings of the 9th Annual ACM Symposium on Computational Geometry (SoCG '93), pp. 173-182.

[17] D. Eppstein, M. Overmars, G. Rote, and G. Woeginger. Finding minimum area *k*-gons. Jur.*Disc. Comp. Geom.* 7(1):45-58, 1992.

[18] F. Preparata and M.I. Shamos. Computational Geometry: An introduction. Springer-Verlag, Berlin, 1985.

[19] Goodman J.E., O’Rourke J. – Handbook of Discrete and Computational Geometry, 2ed, CDC, 2004

International Science Index, Computer and Information Engineering Vol:5, No:5, 2011 waset.org/Publication/5368