Convex Hull Of Imprecise Points Modeled By Segments In The Plane

Ahmad Javad* Ali Mohades* Mansoor Davoodi* Farnaz Sheikhi*

Abstract

Let S be a set of n imprecise points in the plane that each imprecise point is modeled by a segment. In this paper, we study the problem of finding a minimum perimeter convex hull of S that segments are either inside this convex hull or intersected by it. We present the first polynomial time algorithm to solve this problem in $O(n^2 \log n)$ time where the segments are disjoint.

1 Introduction

Computational geometry is a vast area of research that mostly deals with designing algorithms that work with exact input data, but in real world problems, due to devices with limited accuracy, input data might be imprecise. Therefore, a new class of problems focuses on designing algorithms which are able to work with imprecise data. Imprecise data can be modeled by a region they lie on.

Suppose that S is a set of imprecise points where each of its points is modeled by a segment in the plane. We want to find the minimum perimeter convex polygon that has no segments of S outside. We assume there does not exist a line which intersects all segments of S. Goodrich and Snoeyink presented an algorithm that finds a convex polygon whose boundary stabs a set of parallel line segments, in $O(n \log n)$ time [1]. Meijer and Rappaport allowed the interior and the boundary of the polygon to stab the set S of parallel line segments, and found a stabbing polygon of the smallest perimeter, called a $minimumstabbingpolygon$ of S, in $O(nlogn)$ time [4]. Rappaport proposed an algorithm for the problem of computing convex hull of a set of disjoint segments, which is called $minimumpolygontransversals$, in $O(3^k n \log n)$ time [5]. Hassanzadeh showed that algorithm could not work correctly in some cases. So, he corrected it, resulting in an $O(4^k n \log n)$ time algorithm, and presented several approximation algorithms to solve that problem as well [2]. Löffler and van Kreveld studied minimum/maximum perimeter/area convex hull of imprecise points where each imprecise point was modeled by a segment or a square. They also proved the problem of finding maximum area/perimeter convex hull is NP-hard [3].

No polynomial time algorithms are known to solve the problem of finding the minimum perimeter convex hull of a set of segments. In this paper, we present the first polynomial time algorithm which solves this problem in $O(n^2 \log n)$ time.

*Laboratory of Algorithms and Computational Geometry, Department of Mathematics and Computer Science, Amirkabir University of Technology, {ahmadjavas,mohades,mdmonfared,f.sheikhi}@aut.ac.ir

2 Preliminaries

The algorithm that we present to solve the problem proposed, is similar to $QuickHull\ algorithm$ which computes the convex hull of a point set in the plane. Our algorithm is an iterative one that in each iteration computes the convex hull of some special segments of S by using an $unfolding\ method$ [2], and then updates the resultant convex hull regarding the segments that lie outside it.

Before concentrating on details of the algorithm, we present some useful concepts. Let P be a set which includes endpoints of all segments of S, and $CH(P)$ denote its convex hull.

Theorem 1 *Suppose at least one endpoint of each segment of S lies on the boundary of $CH(P)$. A tour MT which visits all segments has the minimum length, if it intersects the segments in their clockwise (or counter clockwise) traversal on the boundary of $CH(P)$.*

Proof. Let $s_1, s_2, , s_n$ be the ordered segments which are visited in clockwise traversal on the boundary of $CH(P)$, and w_i be the intersection point of s_i and MT. Assume MT does not visit segments in their clockwise order, so there exist some segments like s_i that MT visits it after s_j, $(i < j)$. Let MT be $\ldots, w_{i-1}, w_{i+1}, w_{i+2}, \ldots, w_{j-1}, w_j, w_i, w_{j+1} \ldots$ in clockwise order. If $w_i w_j$ intersects $w_{i-1}w_{i+1}$, and $w_{i-1}w_i w_{i+1} w_j$ is a convex quadrilateral, according to the triangle inequality, the tour $\ldots, w_{i-2}, w_{i-1}, w_i, w_{i+1}, \ldots, w_j, w_{j+1}, \ldots$ is a shorter tour, providing a contradiction. Otherwise, if $w_i w_j$ does not intersect $w_{i-1}w_{i+1}$, two cases will be arisen: either $w_{i-1}w_{i+1}$ intersects s_i or does not. If $w_{i-1}w_{i+1}$ intersects s_i, we can obtain a shorter tour by replacing the intersection point with w_i. On the other hand, if $w_{i-1}w_{i+1}$ does not intersect s_i, so s_i certainly lies inside the convex shape (5- or 6-gon) which is constructed with s_{i+1}, s_{i-1} and a part of the boundary

of $CH(P)$. Thus, w_iw_j intersects either s_{i+1} or s_{i-1}. Similarly, in both situations, we can obtain a shorter tour by replacing the intersection point with either w_{i+1} or w_{i-1}, providing a contradiction. □

Lemma 2 *Suppose that at least one point of each segment of S, lies on the boundary of $CH(P)$. The minimum length tour MT which visits all segments of S, is convex.*

Definition 1 *Given a set of ordered segments, a minimum perimeter tour that visits all such segments is denoted by $MTOS$.*

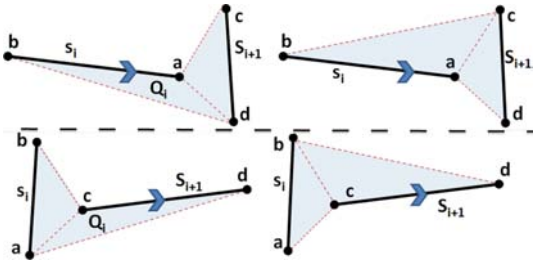

Figure 1: Illustration of two choices that exist to select Q_i.

3 Convex hull of segments algorithm

At first, we compute the convex hull of P and denote it by L_0. Then, by a clockwise traversal on L_0, we find all segments of S which intersect L_0, and insert them into a list called SL_0. Note that if there exist some segments that have more than one intersection point, we insert them only once. We set SL_0 to CS_1, and compute the convex hull of the ordered set CS_1 by using the following method. In this step, we prune segments which are not important in the final solution from CS_1. Based on Theorem 1, MT visits every segment s_i between s_{i-1} and s_{i+1}. Let a_i and b_i be the endpoints of s_i which a_i lies on the convex hull. We will remove s_i from CS_1 if it intersects with $b_{i-1}b_{i+1}$, $a_{i-1}b_{i+1}$ and $b_{i-1}a_{i+1}$. After each removal, we update CS_1 and repeat this process until there are no segments left to remove.

With respect to Theorem 1, for the ordered set CS_1, $MTOS$ should visit s_i before s_{i+1}. Let a and b be endpoints of s_i, and c and d be endpoints of s_{i+1}. Obviously, $MTOS$ crosses either the quadrilateral $abcd$ or $abdc$. Let Q_i be the quadrilateral which $MTOS$ crosses. If either $abcd$ or $abdc$ is a convex quadrilateral, we will select the convex one as Q_i. Otherwise, if both $abcd$ and $abdc$ are non-convex, we will select Q_i as follows. Suppose that the extension of s_i intersects s_{i+1}. Let b be the nearest endpoint of s_i to s_{i+1}. We select the quadrilateral that contains the triangle that lies on the right of the directed segment ab. On the other hand, if the extension of s_{i+1} intersects s_i, and c is its nearest endpoint to s_i, we select the quadrilateral that contains the triangle which lies on the right of the directed segment cd (see Fig. 1).

Each two consecutive quadrilaterals Q_i and Q_{i+1}, which are constructed by the way mentioned , share a segment s_{i+1}. If we start from a segment of CS_1 and cross its related quadrilateral to get the next quadrilateral, we can construct a tour which completely lies inside the union of all quadrilaterals, and also visits all segments of CS_1. This tour is a convex hull for the ordered set CS_1. The minimum tour, which is denoted by CHS_1, could be constructed in linear time by *unfolding method* [2]. See Fig. 2. The process mentioned in this section that computes $MTOS$ is called MTA.

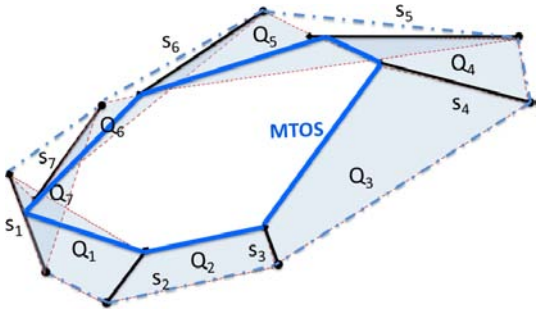

Figure 2: Illustration of consecutive quadrilaterals constructed and $MTOS$ corresponding to CS_1.

Lemma 3 *For a set of n ordered segments, MTA could compute $MTOS$ in $O(n)$ time.*

Suppose that CS_i and CHS_i have been computed up to the i-th iteration. Segments which are located inside CHS_i will be removed form S. Considering segments which are located outside of CHS_i, the convex hull of their endpoints will be computed, and denoted by L_i. With a clockwise traverse on the boundary of L_i, segments which fall outside of L_i and have at least one point on the boundary of L_i, will be added to SL_i in order; note that each segment will be added to SL_i only once. Regarding the location of L_i and CHS_i, CHS_{i+1} could be computed in three different cases as follows.

Case 1: if CHS_i is inside L_i, we will compute intersection points of segments of CS_i with L_i, and with a clockwise traverse on the boundary of L_i, we will add segments of CS_i among those of SL_i, and CS_{i+1} will be computed. Further, convex hull of segments of CS_{i+1} will be computed by using MTA, and denoted by CHS_{i+1}.

Case 2: if CHS_i is outside of L_i (see Fig. 3), there might exist some segments of CS_i which intersect with

L_i at two points (s_1 and s_2, in Fig. 3, are such segments). We will take their farthest intersection points from CHS_i as the place they intersect with L_i (points p_1 and p_2 in Fig. 3). Each segment of this kind will be added to SL_i between the segments of this set which their endpoints locate exactly before and after the specified intersection point of such segment with L_i (in Fig. 3, s_1 (resp. s_2) will be added between s_6 and s_{15} (resp. s_9 and s_8)). Let $s_j, \ldots, s_{j+m}$ be the segments of CS_i which intersect with L_i at two points. The segment of CS_i that is located before s_j (resp. after s_{j+m}) and does not intersect with L_i, is denoted by s_a (resp. s_b), (in Fig. 3, $s_a = s_3$ and $s_b = s_4$). It might be possible that $s_a = s_b$, but the fact that CHS_i is outside of L_i ensures that at least one of these segments exists.

By using an angular sweep line which is along s_a (resp s_b), and rotates around the intersection point of s_a (resp. s_b) and CHS_i, the plane is swept in counter-clockwise (resp. clockwise) direction; the first vertex of L_i that the sweep line intersects with it, is denoted by v_r (resp. v_l), see Fig. 3. The chain of L_i that is located between v_l and v_r in a clockwise traverse, is called *theupperchain*, and denoted by UC. Those segments of CS_i which intersect with L_i at two points, will be removed from CS_i, and the segments of SL_i that at least one of their endpoints is located on UC will be added to CS_i between s_b and s_a in the order that their endpoints are seen in a clockwise traverse on UC, (as an example, in Fig.3; at first, CS_i=$\{s_1, s_2, s_3, s_5, s_4\}$ and after adding the segments of SL_i which one of their endpoints is located on UC, CS_i will be changed into $\{s_4, s_{14}, s_{15}, s_1, s_6, s_7, s_8, s_2, s_9, s_{10}, s_3, s_5\}$). Now, we have an ordered set of segments. By using MTA, we could achieve an optimal convex polygon that intersects the segments of CS_i in order. CHS_{i+1} denotes this polygon, and we set CS_i to CS_{i+1}.

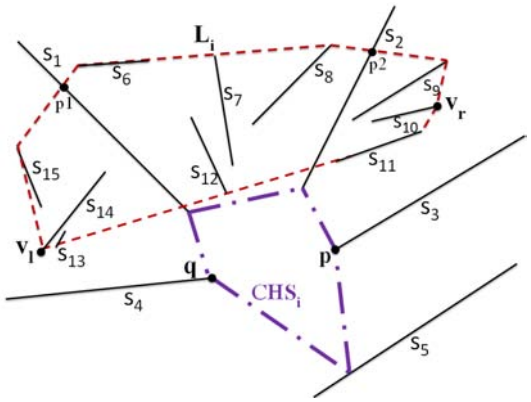

Figure 3: Illustration of case 2.

Theorem 4 *CHS_{i+1} is a convex polygon.*

Proof. Let p (resp. q) be the intersection point of s_a (resp. s_b) and CHS_i. By Theorem 1, we know that segments on UC should be visited in the order they locate on UC, and by Lemma 2, it is clear that the shortest path between p and q which visits those segments, is a convex chain. This chain is denoted by π. π and the chain which exists between p and q in a clockwise traverse on CHS_i, both contain p and q. Therefore, we could construct a polygon T by using them; see Fig. 4. T might have two concave vertices at points p and q. Let q be the concave vertex, and s (resp. r) be its previous (resp. next) vertex in a clockwise traverse on T. Since s_b has been visited before the segments on UC, segment rs intersects s_b. Thus, by substituting rq and qs with rs in T, T still intersects with all segments of CS_{i+1} and its concavity in q is also removed. The same approach could be taken for p, and T becomes a convex polygon which visits all segments of CS_{i+1}, as a result. According to the fact that there exists a convex tour for visiting ordered segments of CS_{i+1}; CHS_{i+1}, which is the output of MTA, will also be convex. □

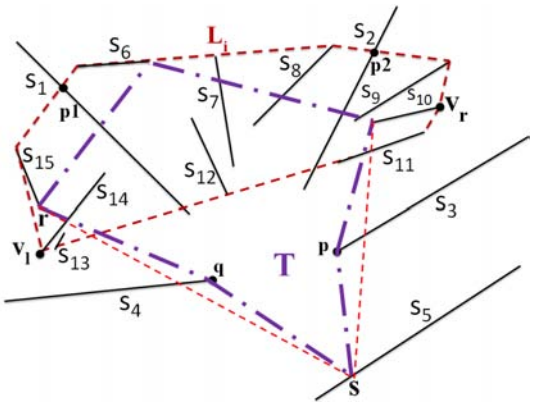

Figure 4: CHS_{i+1} that is computed by MTA is convex.

Case 3: if L_i intersects with CHS_i, then segments which are not completely inside CHS_i will become important in computing CHS_{i+1}; see Fig. 5. In Fig. 5, CS_i=$\{s_1, s_5, s_6, s_2, s_7, s_3, s_8, s_4, s_9\}$) and SL_i=$\{s_{10}, s_{11}, s_{12}, s_{13}, s_{14}, s_{15}\}$. We will find segments of CS_i that intersect with L_i (in Fig. 5, these segments are s_1, s_2 and s_8). Segments of this kind which their intersections with L_i are outside of CHS_i, will be added to SL_i with respect to their intersection points with L_i, and they will be removed from CS_i; in Fig. 5, only s_1 and s_2 are such segments, and after such addition and removal we will have CS_i=$\{s_5, s_6, s_7, s_3, s_8, s_4, s_9\}$ and SL_i=$\{s_{10}, s_1, s_2, s_{11}, s_{12}, s_{13}, s_{14}, s_{15}\}$. We find intersections of L_i and CHS_i. We call each part of L_i that is outside of CHS_i, an exterior chain (denoted by EC). It is clear that all exterior chains are convex

(in Fig. 5, two exterior chains are shown in ovals). Considering each EC, let s_a (resp. s_b) be the segment of CS_i which is visited immediately after (resp. before) the intersection point of EC and CHS_i by a clockwise traverse on the boundary of CHS_i (in Fig. 5, $s_a = s_5$ and $s_b = s_9$ for $EC1$, and $s_a = s_7$ and $s_b = s_6$ for $EC2$). Considering each EC, regarding its s_a and s_b we find a UC. Segments which have at least one of their endpoints on this UC, are added to CS_i between s_a and s_b corresponding to the specified EC. It could be easily proved that segments which are outside of CHS_i and have one of their endpoints on EC, should be visited by CHS_{i+1} between s_a and s_b corresponding to that specific EC (proof is similar to Theorem 1). So, they should be added to CS_i, between s_a and s_b. Adding these segments to CS_i in away that running MTA on CS_i results in a convex polygon, could be done via a similar approach as the one used in case 2; with the only difference that in this case instead of a polygon fallen outside of CHS_i, we have some ECs that segments of them which have one of their endpoints on ECs, should be added to CS_i to achieve CHS_{i+1}. Similar to the proof of Theorem 4, it could be proved that CHS_{i+1}, which is the output of MTA, is also convex in this case, and we set CS_i to CS_{i+1}.

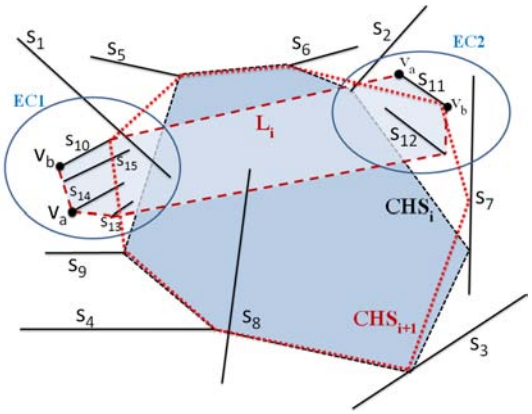

Figure 5: Illustration of case 3.

We will repeat this routine until there are not any segments of S outside of CHS_i at all, and in the end, we will report CHS_i as the final result of the problem.

4 Analysis of the algorithm

In each iteration of the algorithm, L_i should be computed. Finding segments which are outside of CHS_i as well as computing convex hull of their endpoints needs $O(n \log n)$ time in worse case. Regarding the location of L_i and CHS_i, determining the corresponding case could be done in $O(n)$ time. In case 1, According to the fact that L_i is convex and both SL_i and CS_i are ordered, finding intersection of L_i with the segments of CS_i and adding the segments to SL_i could be done in linear time. In case 2, Finding UC and adding the segments which have one of their endpoints on it, takes $O(n)$ time, and finally, in case3, since the total number of segments which have at least one of their endpoints on ECs, is the same as the total number of segments of SL_i which is $O(n)$, and they could be handled in $O(n)$ time in a similar way as case 2. In each iteration $CHS_i + 1$ should be computed by running MTA on CS_i. From the fact that CS_i is always a set of ordered segments, removal of unimportant segments from CSi could be done in $O(n)$ time. Constructing quadrilaterals and utilizing *unfolding method* take linear time [2]. So, MTA runs in $O(n)$ time. Therefore, each iteration of our algorithm takes $O(n \log n)$ time. Since in each iteration just one segments may involve in computing CHS_{i+1} in the worse case, the algorithm runs $O(n)$ times. Thus, the time complexity of the algorithm is $O(n^2 \log n)$ time.

In situations where two consecutive segments of CS_i are collinear, the quadrilateral could not be constructed, but handeling these situations could be done by *unfolding method*, and whenever CHS_i becomes a line segment, it will change into a convex polygon in next iterations. So, these special cases could be handled easily.

5 Conclusion

We have presented the first polynomial time algorithm to compute the convex hull of a set of imprecise points which are modeled by n disjoint segments in the plane. Our proposed algorithm runs in $O(n^2 \log n)$ time. The main idea of the algorithm is to find an order for visiting the segments. We believe that this idea can also be useful for computing the minimum perimeter convex hull of a set of imprecise points which are modeled by polygons instead of segments.

References

[1] M. T. Goodrich, J. S. Snoeyink, *Stabbing parallel segments with a convex polygon.* In Computer vision, Graphics and Image Processing 49, 152170 ,1990.

[2] F. Hassanzadeh. *Minimum Perimeter Convex Hull of a Set of Line Segments: An Approximation.* Master Thesis, Queen's University Kingston, Ontario, Canada, November 2008.

[3] M. Löffler, M. van Kreveld, *Largest and smallest convex hulls for imprecise points.* Algorithmica, 56(2), 235269, 2010.

[4] H. Meijer, D. Rappaport. *Minimum polygon covers of parallel line segments.* Department of Computing and Information Science Technical Report, 90-279, Queens University, 1990.

[5] D. Rappaport, *Minimum polygon transversals of line segments.* Journal of Computational Geometry and Applications, 5, 243256, 1995.