Incremental Convex Hull as an Orientation to Solving the Shortest Path Problem

Phan Thanh An and Tran Van Hoai

Abstract—**The following problem is very classical in motion planning: Let *a* and *b* be two vertices of a polygon and *P* (*Q*, respectively) be the polyline formed by vertices of the polygon from *a* to *b* (from *b* to *a*, respectively) in counterclockwise order. We find the Euclidean shortest path in the polygon between *a* and *b*. In this paper, an efficient algorithm based on incremental convex hulls is presented. Under some assumption, the shortest path consists of some extreme vertices of the convex hulls of subpolylines of *P* (*Q*, respectively), first to start from *a*, advancing by vertices of *P*, then by vertices of *Q*, alternating until the vertex *b* is reached. Each such convex hull is delivered from the incremental convex hull algorithm for a subpolyline of *P* (*Q*, respectively) just before reaching *Q* (*P*, respectively). Unlike known algorithms, our algorithm does not rely upon triangulation and graph theory. The algorithm is implemented by a C code then is illustrated by some numerical examples. Therefore, incremental convex hull is an orientation to determine the shortest path. This approach provides a contribution to the solution of the open question raised by J. S. B. Mitchell in J. R. Sack and J. Urrutia, eds, Handbook of Computational Geometry, Elsevier Science B. V., 2000, p. 642.**

Index Terms—**Motion planning, Euclidean shortest path, convex hull algorithm, convex hull.**

I. Introduction

The problem to determine the Euclidean shortest path between two points in a simple polygon is very classical in motion planning. To date, all methods for solving this problem, as presented in [1], [2], [3], etc, rely on starting with a rather complicated, but linear-time triangulation of a simple polygon. This leads to the open question below raised by J. S. B. Mitchell in [3]: “Can one devise a simple O(n) time algorithm for computing the shortest path between two points in a simple polygon (with n vertices), without resorting to a (complicated) linear-time triangulation algorithm?”.

In 1987, the Steiner's problem of finding the inpolygon of a given convex polygon with minimal circumference was solved completely by the method of orienting curves [4]. In 2008, the method was used to determine the convex hull of a finite set of points in the plane [5]. Efficient algorithms for determining convex ropes in robotics (for determining convex hulls, respectively) were introduced in [6] and [7] ([8], respectively). These problems are variations of the shortest path problem and thus can be solved without resorting to a linear-time triangulation algorithm and without resorting to graph theory.

Geometrically, we determine the shortest path connecting two points a and b that avoids the obstacles - polylines P and Q. Assume without loss of generality that a and b are the first and the final vertices of P and Q, respectively. In this paper, an O($|P||Q|$) time algorithm for determining the shortest path, without resorting to a linear-time triangulation algorithm and without resorting to graph theory, is presented, using the method of incremental convex hull, where $|P|$ ($|Q|$, respectively) is the number of vertices of P (Q, respectively). Under an assumption on links to P and Q, the shortest path consists of the extreme vertices of the convex hulls downward, first advancing on one convex hull formed by vertices of P including a, then on the other formed by vertices of Q, alternating until the vertex b is reached. Each such convex hull is delivered from the incremental convex hull algorithm for a subpolyline of P (Q, respectively) just before reaching Q (P, respectively). Therefore, incremental convex hull is an orientation to determine the shortest path. The algorithm is implemented by a C code and is illustrated by some numerical examples. This paper also provides a contribution to the solution of the Mitchell's open question above.

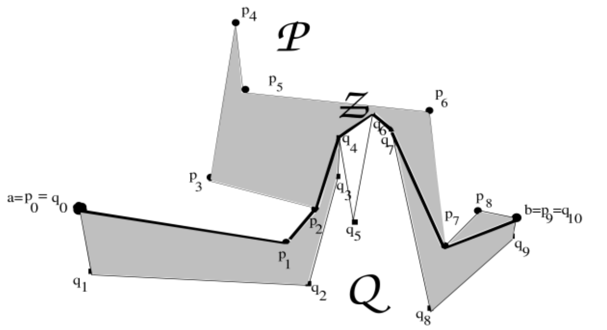

Fig. 1. A simple polygon (shaded) is represented by a counterclockwise polylines P and clockwise Q. Z is the shortest path between a and b (see Fig. 4 and Fig. 5 to know how to find Z).

II. Preliminaries

For a simple polyline $X=<u_0,u_1,...,u_l>$, $[u_0,u_1]$ is called the first edge, u_0 and u_l are called the first and the final vertex of the polyline X, respectively. If $i\leq j$ then we say u_i is before u_j (u_j is after u_i) in the polyline X. u_{i+1} (u_+, respectively) is the next vertex of u_i (u, respectively) and u_{i-1} (u_-, respectively) is the previous vertex of u_i (u, respectively). u is an extreme vertex of the convex set M if $u\in[p,q]\subset M$ implies $u=p$ or $u=q$.

Manuscript received May 10, 2012; revised June 11, 2012.

This work received financial support from Portuguese National Funds through FCT (Fundação para a Ciência e a Tecnologia) under the scope of project Pest-OE/MAT/UI0822/2011, the National Foundation for Science and Technology Development, Vietnam (NAFOSTED) and WB.

Phan Thanh An is with Institute of Mathematics, Hanoi, Vietnam and CEMAT, Instituto Superior Tecnico, Technical University of Lisbon, Portugal (e-mail: thanhan@math.ist.utl.pt).

Tran Van Hoai is with Faculty of Computer Science and Engineering, HCMC University of Technology, Ho Chi Minh City, Vietnam (e-mail: hoai@cse.hcmut.edu.vn).

DOI: 10.7763/IJIEE.2012.V2.181

A simple polygon is represented by two polylines of vertices P and Q such that the first vertices (final vertices, respectively) of P and Q coincide. Two adjacent vertices define an edge of the polygon. Assume that the polygon is given in an orientation as follows: the interior of the polygon lies to the right (left, respectively) as the edges are traversed in the given order of P (Q, respectively) (see Fig. 1). Then we say P (Q, respectively) is a counterclockwise (clockwise, respectively) polyline. Assume without loss of generality that a and b respectively are the first and the final of P and Q. Henceforth, PQ denotes the polygon. We can say that the polygon is formed by polylines P and Q at their first and final vertices a and b, respectively. Furthermore, we assume that the vertices u_i of PQ are supposed to be in general position (no three collinear).

A. Incremental Strategy for Finding Convex Hulls

Many algorithms for determining the convex hull of the polyline $X=<u_0,u_1,...,u_l>$ use a basic incremental strategy. At the j-th stage, they have constructed the convex hull H_{j-1} of the first i vertices $u_0,u_1,...,u_{j-1}$ of X, incrementally add the next vertex u_j, and then compute the next convex hull H_j (see [9]). Most convex hull algorithms construct H_j from H_{j-1} in a similar manner. Namely, they find the right and left tangents from u_j to H_{j-1}, say $[u_j,u^R_j]$ and $[u_j,u^L_j]$, respectively and use these as new edges for H_j in case $u_j \notin H_{j-1}$. From now, u^L_j (u^R_j, respectively) is labelled $\bar{u}_j$ if X is counterclockwise (clockwise, respectively) and we will use the phrase "convex hull" to mean "the set of extreme points of the convex hull".

Many algorithms for determining the convex hull of the polyline $X=<u_0,u_1,...,u_l>$ use a basic incremental strategy. At the j-th stage, they have constructed the convex hull H_{j-1} of the first i vertices $u_0,u_1,...,u_{j-1}$ of X, incrementally add the next vertex u_j, and then compute the next convex hull H_j (see [9]). Most convex hull algorithms construct H_j from H_{j-1} in a similar manner. Namely, they find the right and left tangents from u_j to H_{j-1}, say $[u_j,u^R_j]$ and $[u_j,u^L_j]$, respectively and use these as new edges for H_j in case $u_j \notin H_{j-1}$. From now, u^L_j (u^R_j, respectively) is labelled $\bar{u}_j$ if X is counterclockwise (clockwise, respectively) and we will use the phrase "convex hull" to mean "the set of extreme points of the convex hull".

If there is no vertex of Y in H_{j-1} and there is some vertex v_{j1} of Y in $H_j \backslash H_{j-1}$ (see Fig. 3) then conv$X$ is said to be *firstly intersected* by $v_{j1} \in Y$ between vertices u_{j-1} and u_j (for short, conv$X$ is firstly intersected by Y between u_{j-1} and u_j). Hence, there is at least the vertex v_{j1} of Y trapped in the domain bounded by rays $u_ju^L_j$, $u_ju^R_j$ and the path formed by the convex hull H_{j-1} (before pushing u_j on it). Let Y^* be the set (polyline) of vertices in the domain bounded by rays $u_ju^L_j$, u_ju_{j-1} and the path formed by the convex hull H_{j-1} (Y^* follows the order of Y). Let u^* be a vertex of the polyline (formed by H_{j-1}) from u^L_j to u^R_j (denoted by X^*) and $v^* \in Y^*$ such that apart from u^*, all vertices of X^* are left (right, respectively) to the line u^*v^*. In addition, if there is no vertex of Y^* inside the shaded area $F(u^*,X^*,Y^*)$ formed by the rays u^*v^*, $u_{j-1}u_j$ and the subpolyline $<u^*,u^*_+,...,u^R_j>$ of X^* then $[u^*,v^*]$ is called a *link* to X^* and Y^*. $[u^*,v^*]$ is also called a link to X and Y. u^* and v^* are referred to as link points.

B. Links and Tangent Polylines

We assume that Y is a clockwise (counterclockwise, respectively) polyline forming a simple polyline such that this polyline does not intersect the polyline formed by X and the final vertices of X and Y coincide.

If there is no vertex of Y in Hj for all i≤ j (see Fig. 2) then convX is said to be not intersected by Y before or at the vertex uj.

We define the counterclockwise (clockwise, respectively) *tangent polyline* **TP**(X) of the counterclockwise (clockwise, respectively) polyline X. **TP**(X) is a stack and it allows one to "push" or "pop" on the top of the tangent polyline during the incremental strategy in finding the convex hull of X.

1) Firstly, **TP**(X)=$<u_0,u_1>$.
2) Let the left tangent (right tangent, respectively) from u_j to H_{j-1} be $[u_j,\bar{u}_j]$ ($2\leq j$). For each u_j, if conv$X$ is not intersected by Y before or at u_j then vertices of **TP**(X) after $\bar{u}_j$ are popped and u_j is pushed on **TP**(X).
3) For some vertex u_j of X, conv$X$ is firstly intersected by Y between u_{j-1} and u_j. Let the link to X^* and Y^* be $[u^*,v^*]$ (to find such link will be presented in Section 3.1). Then, all vertices of **TP**(X) after u^* are popped (i.e., if $\bar{u}_j= u^L_j$ ($\bar{u}_j=u^R_j$, respectively) all vertices of the tangent polyline between u^R_j (u^L_j, respectively) and u^* are popped and therefore u^* is the final vertex of **TP**(X).

Assume that $u_0 \in$**TP**(X). For simplicity, we consider the left tangents, counterclockwise X and clockwise Y case only. The right tangents, clockwise X and counter clockwise Y case is considered similarly. The first vertex u_0 of X is an extreme point of conv$X$. In Fig. 1, $[p_2,p_4]$ is the link to $X=P$ and $Y=Q$, $[q_7,p_7]$ is the link to $X=<q_4,...,q_{10}>$ and $Y=<p_2,...,p_9>$ ($q_{10}=p_9$).

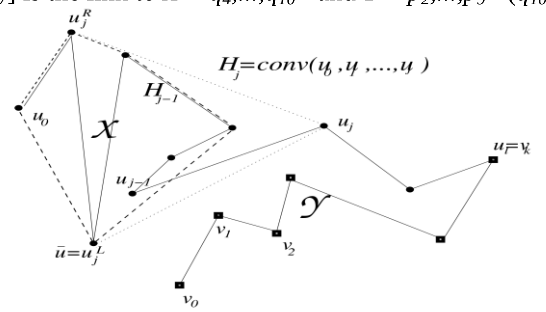

Fig. 2. Conv X is not intersected by Y before or at the vertex uj.

III. The Algorithm

We denote $|X|$ the number of vertices of X and $<x\cup X>$ the polyline delivered from a vertex x and a polyline X (x may not belong to X) such that x is before every vertex of X. For x in X, X_x is delivered from X by discarding vertices before x. Therefore, if $X=<u_0,u_1,...,u_l>$ then $X_{ui}=<u_i, u_{i+1},...,u_l>$ and $<x\cup X_{ui}>=<x,u_i u_{i+1},...,u_l>$. We need the following procedure.

A. Procedure TPL(X,Y) (TPL Stands for Tangent Polyline and Link)

Given counterclockwise polyline $X=<u_0,u_1,...,u_l>$ and clockwise polyline Y such that the final vertex u_l of X and the final vertex of Y coincide and u_0 is an extreme point of convX. TPL(X, Y) finds the tangent polyline **TP**(X) and the link $[u^*,v^*]$ to X and Y, where u* (v*, respectively) is the link point of the polyline X (Y, respectively). It takes |X||Y| time.

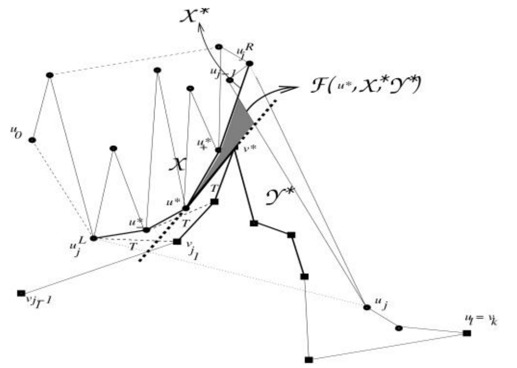

Fig. 3. H_j=conv$X$ is firstly intersected by Y between u_{j-1} and u_j, $v_j \in H_j \backslash H_{j-1}$. [$u^*$,$v^*$] is a link to X^* and Y^* and is called the link to X and Y.

We begin at u_0. At any given stage of the incremental strategy given in Sec. II for determining the convex hull of X, we step along X, examining vertices u_j of the polyline in the order they appear *along* the polyline. Vertices $\bar{u}_j=u^L_j$ and u_j defined by the left tangents are pushed on the tangent polyline **TP**(X).

The tangent polyline **TP**(X) is determined by Section 2.2 a), b) and c)), where conv$X$ is firstly intersected by v_{j1} in Y between vertices u_{j-1} and u_j (see Fig. 3). To do so, we determine if there is some vertex of Y inside the triangle $\bar{u}_j\, u_j\, u_{j-1}$ or on [u_j $\bar{u}_j$] which takes $|Y|$ time. Therefore, the processing necessary to determine if conv$X$ is firstly intersected by Y between vertices u_{j-1} and u_j takes $|X||Y|$ time in the worst case.

Take u^* in X^* and v^* in Y^*. Our search to find whenever [u^*,v^*] is the link to X^* and Y^* has to move on both X^* and Y^*. Note that there is some vertex of Y^* inside $F(u^*,X^*,Y^*)$ iff there is some vertex of Y^* inside the area formed by rays u^*u_{j-1} and u^*v^*. Then this procedure can take O($|X^*||Y^*|$) time.

Since the optimal incremental algorithm takes O($|X|$) time, **TP**(X) and [u^*,v^*] are constructed in O($|X|$)+O($|X||Y|$)+O($|X^*||Y^*|$) time. Hence, **TPL**(X,Y) takes O($|X||Y|$) time. In fact, vertices between $\bar{u}_j$ and u^* on the hull-so-far of X without any vertex of Y inside is maintained in **TP**(X). Moreover, if v^* coincides with the final vertice u_l of X and Y then v^* and vertices of **TP**(X) are extreme vertices of conv$X$.

B. Main Algorithm

The algorithm constructs the shortest path, Z, between a and b in the simple polygon PQ which consists of the set of tangent polylines formed by vertices of P=<a=p_0,p_1, ...,p_n=b> and Q=<a=q_0,q_1,...,q_m=b> (i.e., the set of some extreme edges of the convex hulls of subpolylines of P and Q) and links between these subpolylines. We also use Z to label the polyline formed by ordered vertices of the path Z. Denote V a subpolyline of P (Q, respectively) and V' a subpolyline of Q (P, respectively). If [u^*,v^*] is a link to P and Q, assume that v^* in **TP**(V_{v^*}) for each subpolyline V of P (or Q).

1) Begin at a=p_0=q_0. Set l:=0, V:=P, u^*=p_0 and v^*=q_0.

2) Call **TPL**(V_{u^*},V'_{v^*}) to obtain the tangent polyline **TP**(V_{u^*}) and the link [u^*,v^*] to V_{u^*} and V'_{v^*}. Let Z_l be the path formed by **TP**(V_{u^*}) and <u^*,v^*>. If v^*=b, then Z:=$\bigcup^{j=l}_{j=0} Z_j$, STOP. Else, set l:=l+1 and V:=V'_{v^*}, go to step **2**.

In step **1**, we choose V:=P. This selection (V:=P or V:=Q) is not crucial in the algorithm and does not effect the result. Because each tangent polyline **TP**(X) consists of extreme vertices of the convex hull conv$X$ from u_0 to u^*, Z is determined by the extreme vertices of the convex hulls downward, first advancing on one convex hull formed by vertices of P including a, then on the other formed by vertices of Q, alternating until the vertex b is reached (see Fig. 5).

Example 3.1: Consider the simple polygon PQ in Fig. 1, where P=<a=p_0,p_1,p_2,p_3,p_4,p_5,p_6,p_7,p_8,p_9=b> and Q=<a=q_0,q_1,q_2,q_3,q_4,q_5,q_6,q_7,q_8,q_9,q_{10}=b>. We find the shortest path, Z, between a and b in PQ. Z is determined by left tangent polyline <p_0,p_1,p_2>, right tangent polyline <q_4,q_6,q_7>, left tangent polyline <p_7> and 3 links [p_2,q_4], [q_7,p_7], [p_7,q_{10}]. Z includes the set of the extreme vertices p_0,p_1,p_2 of the convex hull of <p_0,p_1,p_2,p_3,p_4,p_5>$\subset P$, the set of the extreme vertices q_4,q_6,q_7 of the convex hull of <q_4,q_5,q_6,q_7>$\subset Q$, and the set of the extreme vertices p_7,p_9 of the convex hull of <p_7,p_8,p_9>$\subset$P (see Fig. 4).

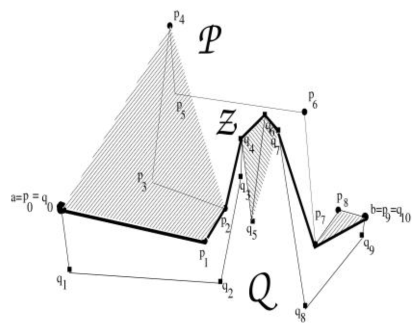

Fig. 4. The shortest path Z between a and b in the simple polygon PQ is determined by the set of the extreme vertices of the shaded convex hulls of subpolylines of P and Q respectively.

C. Correctness of the Algorithm

Lemma 3.1: Assume that Y is clockwise and at the j-th stage in the incremental strategy, the convex hull Hj-1 of the first j vertices u0, u1,..., uj-1 of counterclockwise X is constructed by Section 2.1) and u0 in $TP(X)$. Let [uj,uLj] ([uj,uRj], respectively) be the left tanget (right tanget, respectively) from uj to Hj-1 and X* be the set of vertices of the counterclockwise convex hull Hj-1 from uLj to uRj and u^* in X^* be a link point of a link to X and Y. Assume that u^*+ is the next vertex and u*- is the previous vertex of of u^* in X^* and Y^{**} is the polyline of vertices of Y belonging to the domain bounded by the lines u*u*+,u*-u* and ujuj-1. Then

1) u^* is an extreme vertex of the convex hull of {u^*}$\cup Y^{**}$.

2) The first edge (if viewed from u^*) of the clock-wise convex hull of {u^*}$\cup Y^{**}$ is a link to X and Y. It follows that a link [u^*,v^*] is the first edge of the right tangent polyline (left tanget polyline, respectively) of the polyline <{u^*}$\cup Yv_*$>.

By induction, we conclude that the path, Z, delivered from the algorithm in Section 3.2) determines a set of tangent polylines of subpolyline of P and Q and links between these subpolylines.

Proposition 3.1: The path, Z, delivered from the algorithm in Section 3.2) determines sets of the extreme vertices of the convex hulls downward, first advancing on one convex hull formed by vertices of P including a, then on the other formed by vertices of Q, alternating until the vertex b is reached.

Clearly, the number of these convex hulls is best possible.

Theorem 3.1: The algorithm presented in Section 3.2) computes the shortest path between two vertices a and b in the simple polygon PQ in O(|P||Q|) time.

Proof: First, we prove that Z is the shortest path between two points, a and b, in the simple polygon PQ. Suppose the shortest path is Z^*. Take the links $[u^*_1,v^*_1]$ and $[u^*_2,v^*_2]$ corresponding to Z_l and Z_{l+1}. Then, Z^* intersects with $[u^*_1,v^*_1]$ and $[u^*_2,v^*_2]$ at some z_1 and z_2, respectively.

We now consider a new polyline, Z^*_l, constructed by Z_l and points z_1 and z_2 as follows: z_1 is inserted on the first position and z_2 is pushed on the final position of the polyline Z_l. Thus, $Z^*_l=<z_1,v^*_1>\cup Z_l \cup <u^*_2,z_2>$ and therefore the vertices of Z^*_l between z_1 and z_2 are of Z_l.

By Lemma 3.1 b), for the polyline $<u^*_1,v^*_1>\cup Z_l$, an interior angle made by a vertex and its previous and next vertices is less than π. It follows that this property holds true for the polyline Z^*_l. Furthermore, the length of Z^* between z_1 and z_2 is bigger or equal the length of the path formed by the polyline, which is the path Z between z_1 and z_2 (see Fig. 5). Thus, the path Z^* between z_1 and z_2 coincides with the path Z between z_1 and z_2 and therefore Z^* coincides with Z.

At each step of the algorithm presented in the Section 3, $|V_{u^*}|$ and $|V'_{u^*}|$ are decreasing. In the next step, $V:=V'_{v^*}$. Hence, the algorithm only advances, never backs up, and the number of steps is therefore limited by the number min{$|P|$,$|Q|$}. As shown in Section 3.1, each step takes O($|V_{u^*}||V'_{v^*}|$) time. The worst case occurs when $V_{u^*}=P$ and $V'_{v^*}=Q$ and therefore there is only one link to P and Q. This case takes O($|P||Q|$) time.

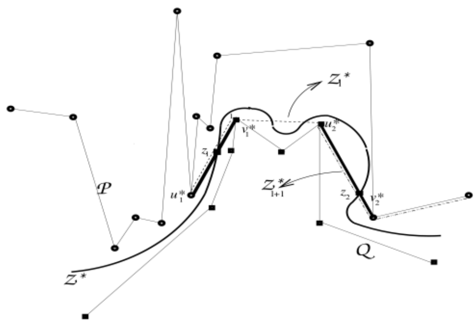

Fig. 5. The path Z^* between z_1 and z_2 should coincide with the path formed by the polyline Z^*_l.

II. IMPLEMENTATION

The algorithm is implemented by a C code, in which the incremental strategy is the Melkman's convex algorithm. The random simple polygon PQ in case $|P|+|Q|=300$ vertices is computed by RPG's "X-Monotone" heuristic in [10] in the square of size 1. a (b, respectively) is the leftmost vertex, i.e. its x coordinate is minimum (rightmost vertex, i.e. its x coordinate is maximum, respectively). The shortest path is presented in Fig. 6.

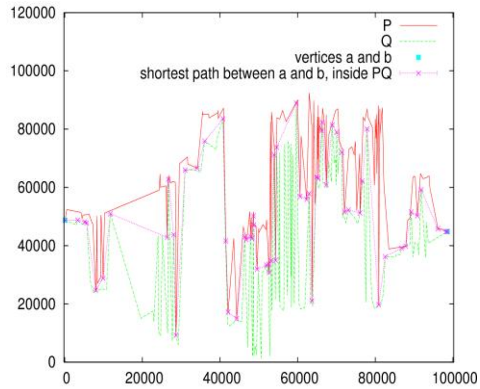

Fig. 6. The shortest path between two vertices a and b of the polygon PQ with the number of random vertices being $|P|+|Q|=300$.

REFERENCES

[1] L. Guibas and J. Hershberger, "Optimal shortest path queries in a simple polygon," *J. Comput. Syst. Sci*, vol. 39, pp. 126-152, 1989.

[2] D. T. Lee and F. P. Preparata, "Euclidean shortest paths in the presence of rectilinear barriers," *Networks*, vol. 14, pp. 393-410, 1984.

[3] J. S. B. Mitchell, "Geometric shortest paths and network optimization," in *Handbook of Computational Geometry*, J. R. Sack and J. Urrutia, eds, Elsevier Science B. V, 2000, pp. 633-701.

[4] H. X. Phu, "Ein konstruktives Lösungsverfahren für das Problem des Inpolygons kleinsten Umfangs von J. Steiner," *Optimization*, vol. 18, pp. 349-359, 1987.

[5] P. T. An, "Method of Orienting Curves for determining the convex hull of a finite set of points in the plane," *Optimization*, vol. 59, no. 2, pp. 175-179, 2010.

[6] P. T. An, "Reachable grasps on a polygon of a robot arm: finding convex ropes without triangulation," *Journal of Robotics and* Automation, vol. 25 (4), pp. 1-7, 2010.

[7] M. A. Peshkin and A. C. Sanderson, "Reachable grasps on a polygon: the convex rope algorithm," *IEEE Journal on Robotics and Automation*, vol. 2, no. 1, pp. 53-58, 1986.

[8] P. T. An, "A modification of Graham's algorithm for determining the convex hull of a finite planar set," *Annales Mathematicae et Informaticae*, vol. 34, pp. 3-8, 2007.

[9] J. O'Rourke, "Computational Geometry in C. Cambridge University Press," *Second Edition*, 1998.

[10] T. Auer and M. Held. "Heuristics for the generation of random polygons," in *Proc.* 8th *Canad. Conf. Computat. Geometry*, Carleton University Press, F. Fiala, E. Kranakis, and J. R. Sack (eds.), Ottawa, Canada, 1996, pp. 38-44.