Send Orders for Reprints to reprints@benthamscience.ae

Open Access

A New Algorithm for the Shortest Path of Touring Disjoint Convex Polygons

Wang Lijuan[1,2], Jiang Bo[*,1], Deng Ansheng[1] and Wei Qi[2]

[1]College of Information Science and Technology, Dalian Maritime University, Dalian, 116026, P.R. China

[2]Dept of Information and Science, Dalian Institute of Science and Technology, Dalian, 116052, P.R. China

Abstract: Given a start point s, a target point t, and a sequence of k disjoint convex polygons in the plane, finding the shortest path from s to t which visits each convex polygon in the given order is our focus. In this paper, we present an improved method to compute the shortest path based on the last step path maps by Dror *et al*. Instead of using of point location in previous algorithm, we propose an efficient method of locating the points in the path with linear query and make the data structures much simpler. Our improved algorithm gives the $O(nk)$ running time which improves upon the time $O(nk\log(n/k))$ by Dror etal., where n is the total number of vertices of all polygons. Furthermore, we have implemented this algorithm by programming. The result shows that our algorithm is correct and efficient..

Keywords: Disjoint convex polygon, last step path maps, linear query.

1. INTRODUCTION

Path planning is one of the central problem areas in computational geometry [1]. The shortest path problem is the most classical example of path planning. Finding the shortest path between two points for the given order objects or obstacles is the main goal [2]. In this paper, we mainly study on the method of computing the shortest path between two points s and t of touring a sequence of disjoint convex polygons given in the plane. The problem can be described as follows.

Given a start point s, a target point t, and a sequence P = (P_1,..., P_k) of simple disjoint convex polygons in the plane. The goal is to find the shortest path that starts from s, visits all convex polygons according to their order and ends at t [3]. In Fig. (**1**), the path linked by bold lines is the shortest path from *s* to *t* which visits the disjoint convex polygons.

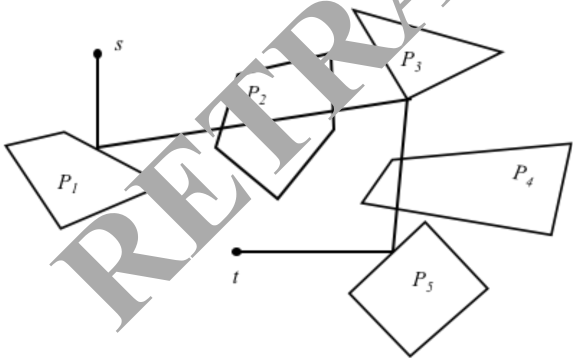


Fig. (1). The shortest path of touring disjoint convex polygons, for k=5.

The problem studied in this paper is a sub-problem of the touring polygons problem (TPP), introduced by Dror, Efrat, Lubiw and Mitchell in STOC '03[4]. Algorithms for solving the touring convex polygons problem have many important applications in many geometric problems, such as the zoo-keeper [5], safari [6], and watchman route problems [7-9]. In the fixed-source safari and zoo-keeper problems, given a start point s in a simple polygon P, and a set of disjoint convex polygons (cages) inside P, each of which having a common edge with P. In the safari problem, we need to seek the shortest route of touring each cages, while in the zoo-keeper problem, the cages can't be allowed to enter, see Fig. (**2**). In the fixed-source watchman route problem, given a simple polygon P and a start point s in it. The goal is to find a shortest route from s such that we can see each point in P from at least one point of the route, see Fig. (**3**). What all these problems have in common is that the shortest visiting path in the given order needed to be found [3]. If the visited order is not specified, it becomes the classical Traveling Salesperson Problem with neighborhoods, which is NP-hard [10].

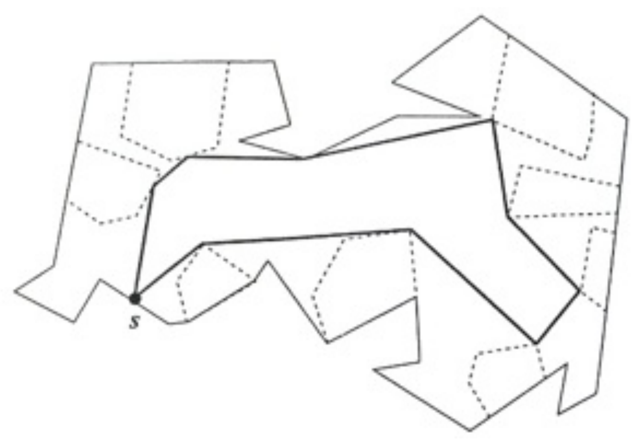


Fig. (2). Safari problem.

*Address correspondence to this author at the College of Information Science and Technology, Dalian Maritime University, Dalian, Liaoning, 116026, P.R. China; Tel: +8618941199928; Fax: +86 41186245057;
E-mail: newdlhot@163.com

1874-4443/15 2015 Bentham Open

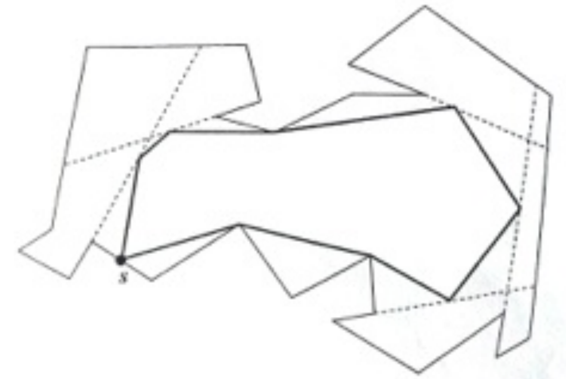


Fig. (3). Watchman route problem.

The touring polygons problem has been intensively studied. Applying the method of last step path maps, Dror *et al.* gave an algorithm running in time $O(nk\log(n/k))$ if the given polygons are disjoint and convex, and an $O(nk^2\log n)$ time if the convex polygons are arbitrarily intersected and the sub-path between any two consecutive polygons is constrained to lie within a simply connected region, where n is the total number of vertices specifying the polygons [4]. They also proved the TPP is NP-hard for the case that they are intersected and non-convex polygons [4, 11]. Arash Ahadi *et al.* also proved that TPP is NP-hard when the polygons are pairwise disjoint [3]. Several approximation algorithms have been proposed for the case of disjoint convex polygons, for example, Fajie Li and Reinhard Klette in 2007 gave an approximate algorithm in $K(\varepsilon)\cdot O(n)$ time by applying the rubber-band algorithm for sequences of convex polygons, where $K(\varepsilon)=(L_0-L)/\varepsilon$, L_0 is an initial path, L that of the true length of the shortest path of convex polygons set P, and n is the total number of vertices of the given polygons [12]. The experiment by Wang and Huo in 2011 showed that the time complexity of rubber-band algorithm is $O(n^2)$ when n is larger [13, 14].

Dror *et al.* has been solved the touring disjoint convex polygons problem using the last step shortest path maps method to compute iteratively [4]. Computing the shortest touring path to a vertex v of P_i ($0 \le i \le k-1$) will be performed at most i-1 point location queries for the polygons P_{i-1}, ..., P_1. It is clearly that a query for point t in the final last step shortest path map determines a query point for the previous last step shortest path map. The last step shortest path map method adopts the point location data structure to save a polygon [4]. Thus, a query for the shortest path from s to any point can be computed in time $O(k\log(n/k))$.

In this paper, instead of computing the point locations independently, we compute the point locations for all the vertices of P_i once by a linear scan on the boundary of P_i, thus, the point location data structure is not needed. This method is simple and can reduce the time complexity of the algorithm by a factor of $O(\log n)$. Thus the $O(nk)$ running time is obtained which improved upon the time $O(nk\log(n/k))$ in the algorithms by Dror etal. The data structure is much simpler and the method of locating the points is more efficient.

2. LOCAL OPTIMAL TOURING PATH

We denote by opt(L) a shortest touring path for the given convex polygons P_1,..., P_k ($i \le k$), i-path a path that starts at s and visits the sequences of P_1,..., P_i ($i \le k$) polygons, and $\pi_i(m)$ a path that starts at s and visits the sequences of P_1,..., P_i ($i \le k$) polygons to m.

We assume that all given convex polygons P_1,..., P_k are simple, and opt(L) visits in order P_1,..., P_k, then the local optimality of opt(L) with respect to the P_1,..., P_k is equivalent to global optimality [4]. Denote by the contact point b of opt(L) with the edge e $e \in P_i(i \le k)$, after opt(L) visits P_1,..., P_{i-1}. There are three cases of OPT(L) contacted with P_i, which are as follows.

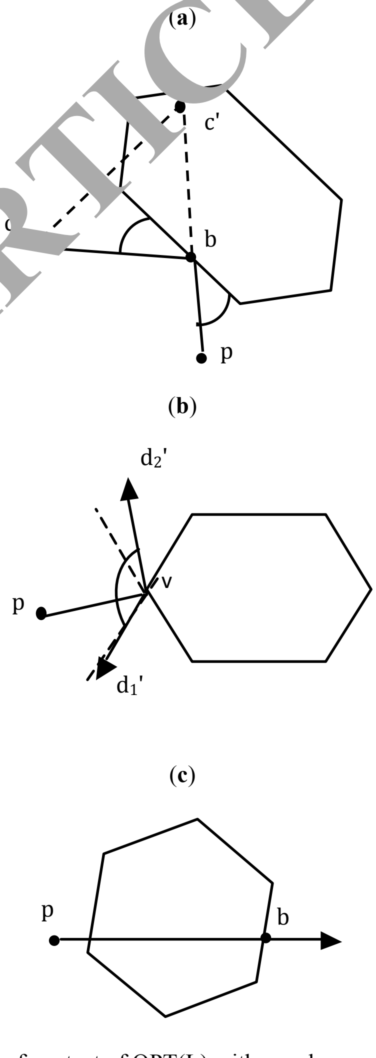


Fig. (4). Types of contact of OPT(L) with an edge e polygon P_i.

Case 1 Edge-reflection contacts

For a bend point b on the interior of an edge e of P_i, OPT(L) makes a reflection contact with an edge e if the angle of OPT(L) coming into the edge e with e (incoming angle) is equal to the angle of OPT(L) going away from e with e (outgoing angle), OPT(L) makes a perfect reflection on the edge e. (see Fig. **4a**), c' is the reflection of c with respect to the line through e.

Case2 Vertex-bending contacts

For a bend point at a vertex $v = e_h \cap e_j (e_h \in P_i, e_j \in P_j)$, consider the rays d_1' and d_2' formed by reflecting the segment $\overline{pv}$ in the two edges, respectively, then, only when the outgoing path segment from v leaves v in the cone ω, which bounded by d_1' and d_2', the local path is the shortest (see Fig. **4b**).

Case 3 Passing-through contacts

OPT(L) passes the edge e through an interior point b of e, we consider b as the second intersection point of the boundary of P_i with OPL(L), and thus OPT(L) reaches the point b from the interior of P_i (see Fig. **4c**).

3. THE LAST STEP SHORTEST PATH MAP

Let G_i be the first contact set of P_i, i.e., the points where the shortest path first reaches a point of P_i after visiting P_1,..., P_{i-1}. G_i is a (connected) chain on the boundary of P_i [4]. In Fig. (**5**), G_1 is the bold edge v_1v_2 and v_2v_3 of P_1. We denote by M_i the last step shortest path map for P_i. Suppose that all given polygons are disjoint, let us compute the first map M_1 as below. For every vertex v of P_1, we first compute the shortest path from s to v. If this path arrives at v from the inside of P_1, then v is not a vertex of G_1, otherwise it is, and $G_1$is a (connected) chain on the boundary of P_1. We can see that OPT(L) may make a reflection on the points and edges of Gi or OPT(L) may go across e. The vertices and edges on the G_1 divide the whole plane into three types which are passing-through regions(C), vertex-bending regions (B) and edge-regions(R).

(i) The passing-through region is bounded by the boundary of G_1 and the extensions of the shortest paths from s to two endpoints of G_1. C region in the Fig. (**5**).

(ii) The edge-reflection region is bounded by one edge e of G_1 and the two rays reflected by the shortest paths from s to the two vertices of e. R(e) region in the Fig. (**5**).

(iii) Let v be a vertex of e, and e' be the other edge incident to v. The vertex-bending region of v is bounded by the two rays which are used in defining the edge-reflection region of e and the edge-reflection or passing-through region of e', which is the triangular region. $B(v_i)$ region in the Fig. (**5**).

Thus, all these subdivision regions in the plane form the last step shortest path map M_1 of P_1.

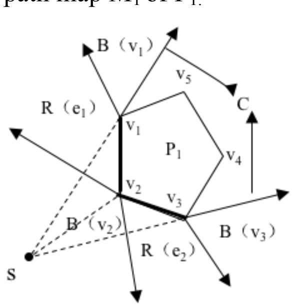


Fig. (5). The last step shortest path maps M_1.

4. THE ALGORITHM

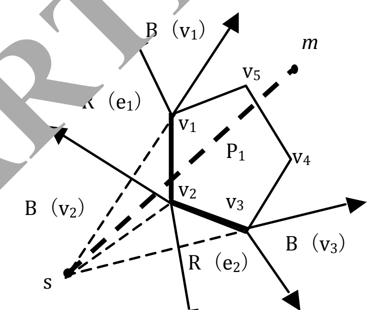
4.1. The Algorithm Shortest Touring Path for Disjoint Convex Polygons

Suppose Given M_1, ..., M_i, we can easily compute a shortest i-path to any query point m as follows.

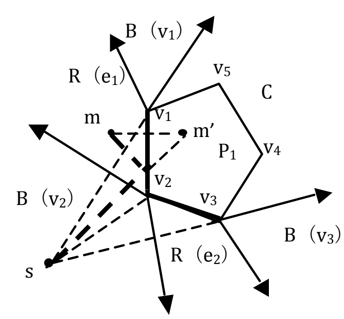
Case 1 m is contained in a passing-through region of M_i. In this case, $\pi_i(m) = \pi_{i-1}(m)$, then we recursively compute the (i-1)-path to m, see Fig. (**6a**).

Case 2 m is contained in an edge-reflection region of M_i. In this case, we let m' be the reflection of m with respect to e, then recursively locate m' in M_{i-1} and compute the (i-1)- path to m', the segment from e to m is the i-1 path to m, see Fig. (**6b**).

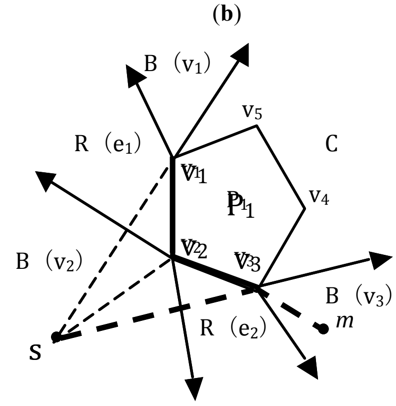
Case 3 m is contained in vertex-bending region of a vertex v. In this case, the last segment of $\pi_i(m)$ is $\overline{vm}$, then we compute the (i-1)-path to v (locating v in M_{i-1}, etc.) recursively, see Fig. (**6c**).



(a)



(b)



(c)

Fig. (6). Three Cases for Computing the Shortest (i-1) Path to m.

We construct each the maps $M_1,..., M_k$ iteratively. The algorithm is follows. Suppose we have obtained the maps $M_1, ..., M_i$, in order to construct the next map M_{i+1}, we first compute the shortest touring paths from s to each vertex v of P_{i+1} as described above. If this path arrives at v from the inside of P_i, then v is not a vertex of G_i. Otherwise it is, the last segment of $\pi_{i-1}(v)$ determines the subdivision of M_{i+1}. Thus, the edge-reflection, passing-through regions and vertex-bending regions of M_{i+1} an be defined analogously. An example of M_2 is shown in Fig. (**7**).

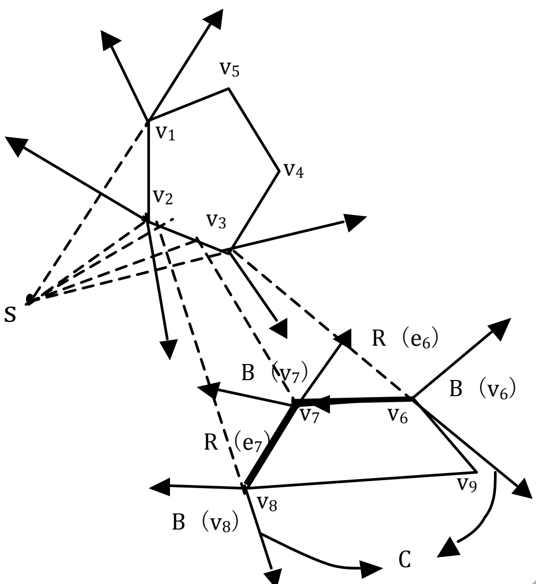


Fig. (7). The Last Step Shortest Path Maps M_2.

Lemma1 Let v' denotes the point in the map M_i which results from the vertex v of P_{i+1}, called mapping-point of v. Then the edges consist of the points sequences v_i' (i=1,.., m, m=$|p_{i+1}|$) in the map M_i which results from the vertices P_{i+1} form at most three edge chains along the boundary of P_i.

Proof. Let the edges in the passing-through region as one edge chain because of the way of locating points is same, the edges in the other zones is divided into two cases according to the trend of rising or falling. Thus, for each convex polygon, the edges on the boundary is formed at most three edge chains, as shown in Fig. (**8**). Let a and b be two vertices of P_{i+1}, and a' and b' be their mapping-points in M_i. The last portions of two shortest touring paths (i.e., from a' to a and from b' to b) can't cross in Mi, except for the following situations: (i) a' and b' are possible the same point, due to the vertex-bending contacts of the shortest touring paths to the vertices of P_{i+1}; or (ii) one path completely is contained by the other. A shortest i-path to any query point m is iteratively computed by the above mentioned method. So, if the point a moves along the boundary of P_{i+1}, the shortest path points on P_i will form two or three edge chains, which depends on the position of starting point in P. For example, the mapping-points in P_1 which result from the vertices of P_2 only form one edge chain, the bold line is the edge chain in Fig. (**7**).

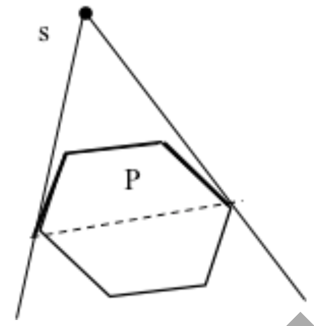


Fig. (8). The three Edge Chains of Polygon P.

Lemma2 Given $M_1,..., M_i$, the path $\pi_i(q)$ can be determined in time $O\left(\sum_{j=1}^{i}|p_j|\right)$, and the map M_{i+1} can be constructed in time $O\left(\sum_{j=1}^{i}|p_j|+i|p_{i+1}|\right)$.

One can easily see that it needs $O(|p_i|)$ time to locate the shortest path point in the M_i for m, and all the maps $M_1,..., M_i$ needed to be visited. The time to find $\pi_i(q)$ is then $O(|p_i|+|p_{i-1}|+\cdots|p_1|)$, denoted by $O\left(\sum_{j=1}^{i}|p_j|\right)$. To construct M_{i+1}, for each vertex v of P_{i+1}, we compute $\pi_i(v)$. For example, we first compute $\pi_i(v_1)$, and the time to find $\pi_i(v_1)$ is $O\left(\sum_{j=1}^{i}|p_j|\right)$. According to Lemma1, locating all the other shortest path points in M_i, for the other vertices of P_{i+1} can be done by a constant number of linear scans in M_i, and it needs $O(i|p_{i+1}|)$ time. So the map M_{i+1}can be constructed in time $O\left(\sum_{j=1}^{i}|p_j|+i|p_{i+1}|\right)$.

Each map M_i can be constructed iteratively, so all maps $M_1,..., M_{k+1}$ can be computed in $O\left(\sum_{i=1}^{k-1}\left(\sum_{j=1}^{i}|p_j|+i|p_{i+1}|\right)\right)$, since $O\left(\sum_{i=1}^{k-1}(i|p_{i+1}|)\right)=O\left(k\sum_{i=1}^{k-1}(|p_{i+1}|)\right)=O(kn)$, and $O\left(\sum_{i=1}^{k-1}\left(\sum_{j=1}^{i}|p_j|\right)\right)=O\left(\sum_{i=1}^{k-1}n\right)=O(kn)$. Thus, all maps can be computed in $O(kn)$ time.

Therorem1 The Touring polygons problem for k disjoint convex polygons with input size n, a data structure of size $O(n)$ can be built in time $O(kn)$ that enables shortest i-path queries to any query point m to be answered in time $O(n)$, where n is the total number of vertices of all the polygons.

4.2. The Implementation of Algorithm

The algorithm presented in this paper has been implemented by program, the division of the whole plane is shown in Fig. (**9**), and the running result of the shortest path is shown in Fig. (**10**). To make the result clearly visible, we only present the result of 5 disjoint convex polygons. This example has contained edge-reflection contacts, vertex-bending reflection contacts, and passing-through contacts three cases of OPT(L) contacted with P_i mentioned above.

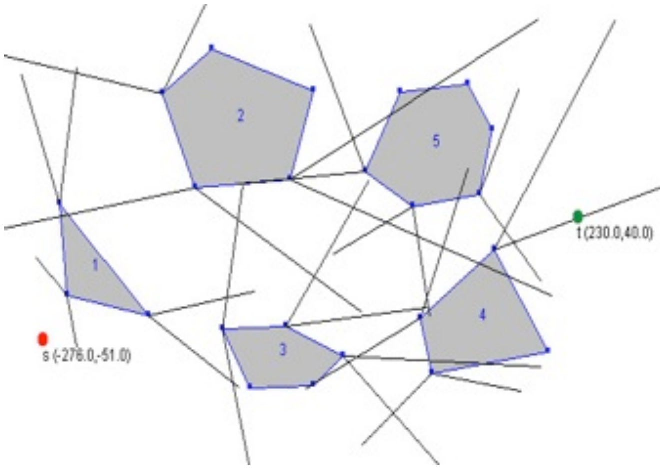


Fig. (9). The Last Step Shortest Path Map M_i for P_i (i=1,…, 5).

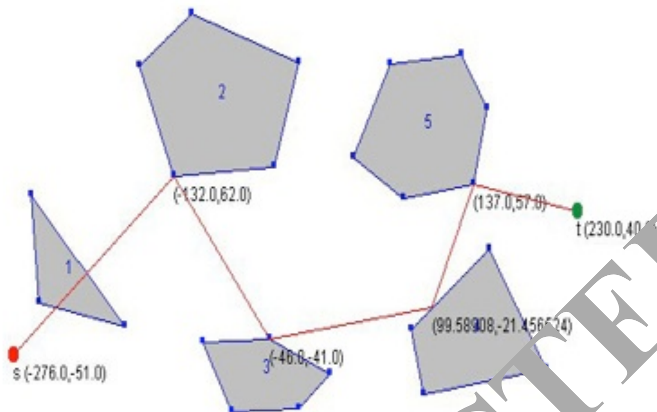


Fig. (10). The Running Result of Disjoint Convex Polygons, for k=5.

CONCLUSION

In this paper, we present an fast algorithm of locating the path points in computing the shortest path of touring a sequence of disjoint convex polygons and we give an $O(kn)$ time solution, where k is the number of polygons and n is the total number of vertices of the polygons. Our results improve upon the previous time $O(nk\log(n/k))$.

This research has made preliminary results. A more efficient time solution to the problem of touring disjoint and convex polygons problem is an open problem. In addition, finding the shortest path of touring the convex polygons possibly intersected is also our further study.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

This work is supported by the National Natural Science Foundation of China (No.61173034, 61272171), General Project of Liao Ning Province Science and Research (No. L2015105), and the General Project of Dalian Institute of Science and Technology (No. KYZ1416).

REFERENCES

[1] Hakan Jonsson, "The Traveling Salesman Problem for lines in the plane," *Information Processing Letters*, vol.82, pp.137-142, 2002.

[2] Mark de Berg, Otfried Cheong, Marc van Kreveld, *et al.,* Computational Geometry Algorithms and Appli- cations (Third Edition), Springer-Verlag Publishers: Berlin, pp. 10-11, 2009.

[3] Arash Ahadi, Amirhossein Mozafari, Alireza Zarei, "Touring a sequence of disjoint polygons: Complexity and extension", *Theoretical Computer Science*, vol.556, pp. 45-54, 2014.

[4] M. Dror, A. Efrat, A. Lubiw and J.S.B. Mitchell, "Touring a sequence of polygons," *in Proc. 35th Annu. ACM Sympos. Theory Comput.*, California, USA, pp. 473-482, 2003.

[5] John Hershberger, Jack Snoeyink, "An efficient solution to the Zookeeper's problem", *in: CCCG*, pp.104–109, 1994.

[6] Xuehou Tan, Tomio Hirata, "Shortest safari routes in simple polygons", *in: Algorithms and Computation, Springer*, Berlin, Heidelberg, pp. 523–530, 1994.

[7] Adrian Dumitrescu, Joseph S. B. Mitchelland Paweł Żyliński, "Watchman Route for Line and segments", *Algorithm Theory – SWAT 2012 Lecture Notes in Computer Science*, vol .7357, no. 14, pp. 36-47, 2012.

[8] Xuehou Tan, Tomio Hirata, "Constructing shortest watchman routes by divide-and-conquer", *in: Algorithms and Computation*, Springer, Berlin, Heidelberg, pp. 68–77, 1993.

[9] Xuehou Tan, "A linear-time 2-approximation algorithm for the watchman route problem for simple polygons", *Theoret. Comput. Sci,* vol.384, no.1, pp.92–103,2007.

[10] Adrian Dumitrescu, "The traveling saleman problem for lines and rays in the plane", *The proceeding of the 22nd Canadian Conference on Computational Geometry(CCCG2010)*, Cannada, pp.257-260,2012.

[11] A. Ahadi, A. Mozafari and A. Zarei, "Touring disjoint polygons problen is NP-hard", *in Proc. Of COCOA'2013*, LNCS 8287, pp.351-360, 2013.

[12] LI Fa-jie and KLETTE Reinhard, "Shortest path algorithms for sequences of polygons", *CAAI Transactions on Intelligent Systems*, vol.3, no.1, pp.23-30, 2008.

[13] Lijuan Wang, Li Huo, and Dandan He, "An Improved Algorithm for Euclidean Shortest Paths of Visiting Line Segments in the Plane", *Journal of Convergence Information Technology*, vol.6, no.6, pp.119-125,2011.

[14] Lijuan Wang, Bo Jiang, Ansheng Deng, Qi Wei, "Fast Computation of Shortest Path for Visiting Segments in the Plane", *The Open Cybernetics & Systemics Journal*, vol.8, no.4, pp.224-229,2014.

Received: May 26, 2015 Revised: July 14, 2015 Accepted: August 10, 2015