Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE @ DIRECT®

ELSEVIER

Information Processing Letters 87 (2003) 179–186

Information Processing Letters

www.elsevier.com/locate/ipl

# Finding shortest safari routes in simple polygons

Xuehou Tan [a,*], Tomio Hirata [b]

[a] *School of High-Technology for Human Welfare, Tokai University, 317 Nishino, Numazu 410-0395, Japan*
[b] *Faculty of Engineering, Nagoya University, Chikusa-ku, Nagoya 464-8603, Japan*

Received 4 March 2002; received in revised form 7 March 2003

Communicated by F.Y.L. Chin

**Abstract**

Let $P$ be a simple polygon, and let $\mathcal{P}$ be a set of disjoint convex polygons inside $P$, each sharing one edge with $P$. The *safari route problem* asks for a shortest route inside $P$ that visits each polygon in $\mathcal{P}$. In this paper, we first present a dynamic programming algorithm with running time $O(n^3)$ for computing the shortest safari route in the case that a starting point on the route is given, where $n$ is the total number of vertices of $P$ and polygons in $\mathcal{P}$. (Ntafos in [Comput. Geom. 1 (1992) 149–170] claimed a more efficient solution, but as shown in Appendix A of this paper, the time analysis of Ntafos' algorithm is erroneous and no time bound is guaranteed for his algorithm.) The restriction of giving a starting point is then removed by a brute-force algorithm, which requires $O(n^4)$ time. The solution of the safari route problem finds applications in watchman routes under limited visibility.

*Keywords:* Computational geometry; Safari routes; Zookeeper's routes; Adjustments

## 1. Introduction

Shortest paths are of fundamental importance in robotics and computational geometry. The *safari route problem*, introduced by Ntafos [9], is defined as follows: given a simple polygon $P$ and a set $\mathcal{P}$ of disjoint convex polygons inside $P$, each sharing one edge with $P$, find a shortest path inside $P$ that visits each polygon in $\mathcal{P}$. One may consider it as minimizing travel for hunting animals in enclosures. If we require that the route should not enter the interior of any polygon in $\mathcal{P}$, then it introduces the *zookeeper's route problem*, where the route is designed for a zookeeper to feed animals [4,7]. See Fig. 1 for an example, in which a starting point $s$ on the route is given (the shortest safari route is drawn in solid line and the shortest zookeeper's route is drawn in dotted line).

The solution of the safari route problem finds applications in watchman routes under limited visibility [9]. The *watchman route problem* deals with finding a shortest closed route in a simple polygon $P$ such that every point in the interior of $P$ can be seen from at least one point along the route [2,12]. If the visibility of the watchman is limited to a distance $d$, then we have two variants of the watchman route problem. The *$d$-watchman route problem* asks for a shortest route such that each point in the boundary of $P$ is $d$-visible (i.e., visible and at most $d$ away) from some

* Corresponding author.
*E-mail address:* tan@wing.ncc.u-tokai.ac.jp (X. Tan).

0020-0190/$ – see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0020-0190(03)00284-9

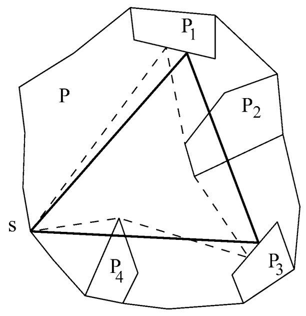

Fig. 1. Shortest safari and zookeeper's routes.

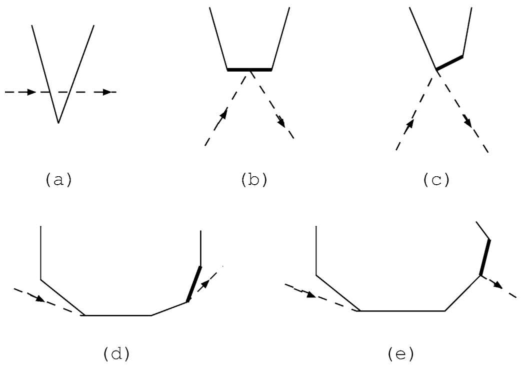

Fig. 2. Types of contact of a safari route with polygons in $\mathcal{P}$.

point along the route, and the *$d$-sweeper route problem* asks for a shortest route such that each point of $P$ is $d$-visible from the route. Based on the solution to the safari route problem, approximation algorithms for these two problems are given in [9].

In Section 2 of this paper, we first define the *adjustments* to a safari route. An adjustment produces a new and shorter route, but it may be the case that the resulting route is not a safari one. A *recovery test* is then designed to check out a non-safari route and recover it back to the safari route. In Section 3, we present an $O(n^3)$ time algorithm for computing the shortest *fixed* safari route, i.e., the route is forced to pass through a starting point $s$ on the boundary of $P$, where $n$ is the total number of vertices of $P$ and polygons in $\mathcal{P}$. Our algorithm first finds an initial safari route and then repeatedly adjusts the current route until it becomes the shortest one. Dynamic programming is used so that no adjustments performed by our algorithm can be undone. (Ntafos also claimed an $O(mf(n))$ time algorithm for computing the shortest fixed safari route [9], where $m$ is the cardinality of $\mathcal{P}$ and $f(n)$ is the time required to compute the shortest fixed zookeeper's route [4,7]. However, the time analysis of his algorithm is erroneous. See Appendix A for details.) Section 4 removes the restriction of forcing the route through $s$ and gives an $O(n^4)$ time solution to the safari route problem.

## 2. Adjustments and recovery tests

Let $P_1, \ldots, P_m$ be the order of the polygons in $\mathcal{P}$ indexed in a clockwise scan of the boundary of $P$. Without loss of generality, we assume that no common tangent segment of $P_i$ and $P_{i+1}$, having $P_i$ and $P_{i+1}$ in the same side and towards the interior of $P$, intersects with the boundary of $P$ and no common tangent segment of $P_{i-1}$ and $P_{i+1}$ intersects with $P_i$ (otherwise, the given $P$ and $\mathcal{P}$ can be so preprocessed, see [9]).

Consider types of contact of a safari route with polygons in $\mathcal{P}$. We say a route *crosses* a convex polygon of $\mathcal{P}$ if it intersects the interior of the polygon (Fig. 2(a)). A route *reflects* on a polygon if it has a single point contact with the polygon at which the route turns right when one follows $R$ in the clockwise direction (Fig. 2(b)–(c)). We refer to the *incoming* (*outgoing*) *angle* of a route $R$ with respect to a reflected edge $e$ as the angle between $e$ and the segment of route $R$ coming into (moving away from) $e$. A reflection contact is *perfect* if the incoming angle of the reflection is equal to the outgoing angle. Usually, a perfect reflection contact occurs at an interior point of the reflected edge (Fig. 2(b)), while a non-perfect one occurs at a vertex of the polygon (Fig. 2(c)). A route *wraps* around a polygon if its contact with the polygon is a chain of zero or more edges (Fig. 2(d)–(e)). A wrap contact is *complete* if it corresponds to left turns at all wrapped vertices belonging to the overlapped chain (Fig. 2(d)); otherwise, it is incomplete (Fig. 2(e)). Observe that a shortest safari route should reflect on some polygons in $\mathcal{P}$ and crosses others, while a shortest zoo-keeper route should reflect on some polygons in $\mathcal{P}$ and wrap around others (see also Fig. 1).

The following lemma gives us a method to design a safari route algorithm, that is, the polygons in $\mathcal{P}$ should be visited clockwise or counterclockwise.

(Note that the proof given in [9] is actually independent of the starting point $s$.)

**Lemma 1** (Ntafos [9]). *Let $P_1, \ldots, P_m$ be the order of the polygons in $\mathcal{P}$ indexed in a clockwise scan of the boundary of $P$. Then there is a shortest safari route that visits the polygons of $\mathcal{P}$ exactly in that order.*

A safari route is characterized by the set of the polygons, called *active polygons*, with which the route makes reflection or wrap contacts. Specially, we call the edges where the route leaves from active polygons or immediately left to the points where the route leaves from, *active edges*. See also Fig. 2 for examples, where active edges are drawn in bold lines.

Given a set of active polygons, we can compute its optimal route by the unfolding method [2]. Let $\mathcal{P}'$ denote the set of active polygons ($\mathcal{P}'$ is a subset of $\mathcal{P}$), and let $P - \mathcal{P}'$ denote the portion of $P$ where all polygons in $\mathcal{P}'$ are removed from $P$. First, the interior of $P - \mathcal{P}'$ is triangulated. The triangulation is then unfolded using active edges as mirrors, in the order of their polygons, and finally the shortest path between the starting point $s$ and its image $s'$ in the unfolded polygon is computed and this shortest path is folded back to obtain the desired safari route. The time taken by this process is linear to $n$ since a simple polygon can be triangulated in linear time [1] and the shortest path between two points in a triangulated polygon can also be found in linear time [6]. In the case that point $s$ is not given, we are faced with the problem of finding the shortest path between a starting edge and its image in the unfolded polygon. Also, a solution of this problem has been reported in the literature. In [5], a linear time algorithm is given for the problem of finding a shortest *aquarium keeper's route* that visits all edges of a simple polygon, which is reduced to the problem of finding a shortest path between an edge and its image in the unfolded polygon.

Suppose that a safari route $R$ is computed using the unfolding method described above. Two types of adjustments may make to the route $R$. Removing any completely wrapped polygon from the set $\mathcal{P}'$ of active polygons produces a shorter safari route $R'$. See Fig. 3(a). We call such an operation the *crossing adjustment*. Route $R$ may also be shortened by changing its set of active edges. Assume that the vertices of $P$ are indexed in the clockwise order

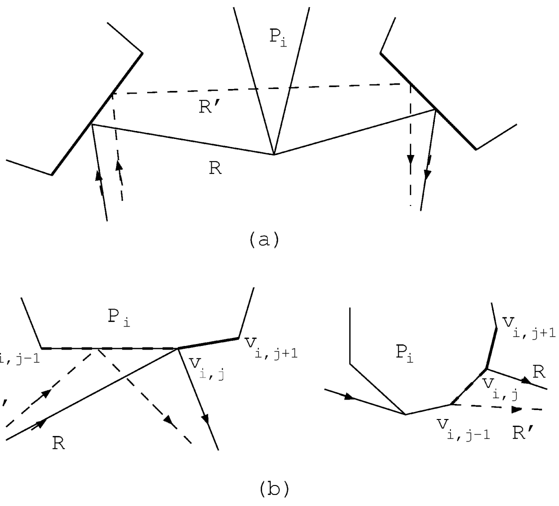

Fig. 3. (a) Crossing adjustments; (b) Shift adjustments.

(starting at a vertex, say, $s$), and the vertices of a polygon $P_i$ of $\mathcal{P}$ in the counterclockwise order, starting and ending at the vertices of $P_i$ lying on the boundary of $P$. Assume that $R$ reflects at a vertex $v_{i,j}$ of a polygon $P_i$. Route $R$ is adjustable at $v_{i,j}$ towards $v_{i,j-1}$ if and only if the incoming angle of $R$ with the reflected edge $(v_{i,j-1}, v_{i,j})$ is smaller than the outgoing angle. In this case, the active edge of $P_i$ should be changed from $(v_{i,j}, v_{i,j+1})$ to $(v_{i,j-1}, v_{i,j})$. We call it the *shift adjustment*. See Fig. 3(b). Note that a shift adjustment occurs at any non-complete wrap contact, as the incoming angle of $R$ with the edge $(v_{i,j-1}, v_{i,j})$ is zero (see also Fig. 3(b)). The condition for $R$ to be adjustable at $v_{i,j}$ towards $v_{i,j+1}$ can be given analogously. If $R$ is adjustable at $v_{i,j}$ towards $v_{i,j-1}$ (respectively $v_{i,j+1}$), we say the *adjustable direction* on $P_i$ is clockwise (respectively counterclockwise). Route $R$ is called *one-place-adjustable* if only a crossing or shift adjustment to $R$ is possible.

Observe that an adjustment implies a call of the unfolding procedure for the new set of active edges, that is, the new route $R'$ shown in Fig. 3 should be computed using the unfolding method. However, the above adjustments do not assure that all polygons in $\mathcal{P}$ are visited by the route $R'$. Sometimes it may be the case that some polygons of $\mathcal{P}$, which were previously crossed, are not visited by $R'$. Thus, after an adjustment is done, we have to check whether route $R'$ intersects all polygons in $\mathcal{P}$ or not. If not, we should

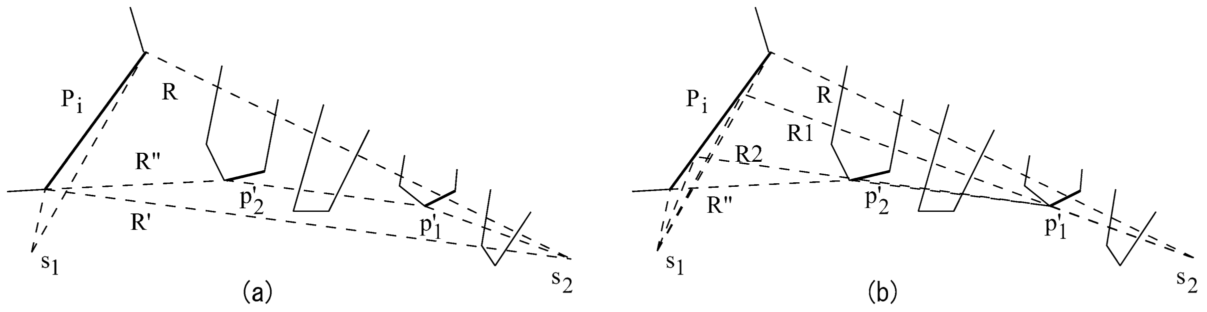

Fig. 4. Recovery tests.

recover it back to the safari route. Since route $R'$ visits the polygons of $\mathcal{P}$ in the clockwise order, the polygons not intersecting with $R'$ can simply be found in linear time.

Assume that route $R'$ is obtained after a shift adjustment on polygon $P_i$ is made, but not a safari one. See Fig. 4(a) for an example. (A crossing adjustment can be considered as two shift adjustments on the active polygons adjacent to the wrapped polygon. See Fig. 3(a).) Let $s_1$ and $s_2$ denote two common points of $R$ and $R'$ such that there are no more common points in the portions of $R$ and $R'$ from $s_1$ to $s_2$. We have to select some from the missed polygons in $R'$ so that making the chosen polygons active leads to a route that crosses others. Let $P_x$ denote a polygon that is not visited by $R'$, and let $p_x$ denote the vertex of $P_x$ such that the line passing through $p_x$ and $s_1$ when $x < i$ (or $s_2$ when $x > i$) is tangent to $P_x$. Then we find the vertex, denoted by $p'_1$, whose tangent line intersects with the active edge of $P_i$ at furthest to the reflection point of $R'$ on $P_i$ among these vertices $p_x$. If two shortest paths, one from $s_1$ to $p'_1$ and the other from $p'_1$ to $s_2$, cross the rest of the missed polygons, then we are done. Otherwise, the above process is done again with $s_1$ (or $s_2$) $= p'_1$ if $s_1$ (or $s_2$) and $p'_1$ determine the tangent line. The found vertices $p'_j$ together with $s_1$ and $s_2$ form a convex chain that reflects on the polygons containing vertices $p'_j$ and crosses other missed polygons (Fig. 4(a)). Hence, making the polygons containing vertices $p'_j$ active gives a new safari route $R''$. Note that $|R''| \leqslant |R|$ (but $|R''| \geqslant |R'|$ as $R'$ is not a safari route), where $|X|$ denotes the length of a route $X$. In the example shown in Fig. 4(b), the incoming angle of $R'$ or $R$ with the active edge of $P_i$ is smaller than the outgoing angle. So we have $|R| \geqslant |R1| \geqslant |R2| \geqslant |R''|$, where $R1$ (respectively $R2$) is obtained by extending the segment of $R''$ adjacent to $p'_1$ (respectively $p'_2$). We call such an operation the *recovery test*. Clearly, it requires linear time to make a polygon active. If the number of the polygons made active by a recovery test is $m_i$, the total time taken is $O(m_i n)$.

## 3. Computing the shortest fixed safari route

In this section, we present a dynamic programming algorithm for computing the shortest fixed safari route. To this end, we define a route from the starting point $s$ to a vertex $v_l$ of polygon $P_l$ as the *partial safari route to* $v_l$ if it lies in the region $P - P_l$ (i.e., without entering $P_l$) and visits either the set of polygons $P_1, \ldots, P_{l-1}$ or the set of polygons $P_{l+1}, \ldots, P_m$.

**Lemma 2.** *There is a unique non-adjustable safari route that is forced to pass through a starting point $s$ on the boundary of $P$.*

**Proof.** It can simply be proved by an induction on the number $m$ of the edges reflected by the shortest partial safari route from a source point $x$ to a target point $y$ ($x$ may be $y$ itself). Since the proof is almost the same as that given for the watchman route, we refer it to [12]. (Alternate proofs can also be found in [3,9].) □

**Lemma 3.** *A fixed safari route $R$ is the shortest one if and only if $R$ is non-adjustable.*

**Proof.** It can be shown by an argument similar to the proof of [3, Corollary 1]. □

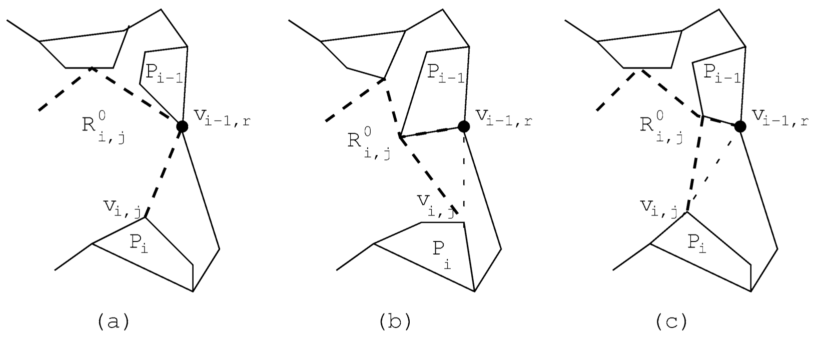

Fig. 5. Possible configurations between $R^0_{i,j}$ and $P_{i-1}$.

Consider the starting point $s$ as an additional polygon $P_{m+1}$. Our algorithm computes the shortest partial safari routes to all vertices of the polygon $P_i$, in the increasing order of index $i$, using the previously obtained partial routes. The final route from $s$ to $P_{m+1}$ ($= s$) is obviously the shortest fixed safari route.

First, the shortest paths in the region $P - P_1$ from $s$ to the vertices of $P_1$ are the shortest partial safari routes to the vertices of $P_1$. Assume now that we have obtained the shortest partial safari routes to all vertices of $P_1, P_2, \ldots, P_{i-1}$. To compute the shortest partial safari route to a vertex $v_{i,j}$ of $P_i$, we first find an initial one-place-adjustable route $R^0_{i,j}$. Let $R_{i-1,r}$ denote the shortest partial safari route to the rightmost vertex $v_{i-1,r}$ of $P_{i-1}$ (i.e., index $r$ is the maximum among the vertices of $P_{i-1}$). If route $R_{i-1,r}$ does not overlap with the boundary of $P_{i-1}$, then route $R^0_{i,j}$ is formed by combining $R_{i-1,r}$ with the shortest path between $v_{i-1,r}$ and $v_{i,j}$. See Fig. 5(a). (Remember that route $R_{i-1,r}$ cannot enter polygon $P_{i-1}$.) If route $R_{i-1,r}$ overlaps with the boundary of $P_{i-1}$, then we compute the shortest paths from $v_{i,j}$ to $v_{i-1,r-1}, \ldots, v_{i-1,r'}$ ($r' < r$) until the route $R^0_{i,j}$, combining the shortest partial safari route $R_{i-1,r'}$ with the shortest path from $v_{i-1,r'}$ to $v_{i,j}$, makes a complete wrap with $P_{i-1}$ (Fig. 5(b)) or a reflection contact with $P_{i-1}$ (Fig. 5(c)). Clearly, the obtained route $R^0_{i,j}$ is either one-place-adjustable or non-adjustable.

If route $R^0_{i,j}$ is non-adjustable, then it is the shortest partial safari route to $v_{i,j}$ and we are done. Otherwise, we perform the only adjustment to $R^0_{i,j}$ and a recovery test later. Let $R^1_{i,j}$ denote the resulting safari route. Generally, route $R^1_{i,j}$ may be adjustable on several polygons. We do not perform any adjustment to $R^1_{i,j}$. Instead, we take out the shortest partial safari route to the vertex of polygon $P_l$, where the last adjustment (i.e., index $l$ is the biggest among the adjustable polygons) to $R^1_{i,j}$ is possible, and concatenate it to the rest part of $R^1_{i,j}$ from the vertex of $P_l$ to $v_{i,j}$. Let $R^2_{i,j}$ denote the resulting route. If $R^2_{i,j}$ is non-adjustable, then we are done. Otherwise, it is adjustable only on $P_l$. The only adjustment to $R^2_{i,j}$ and a recovery test are then performed. In this way, we can eventually find the shortest partial safari route to $v_{i,j}$.

The following lemma states that no adjustments performed by our dynamic programming algorithm can be undone.

**Lemma 4.** *Let $R^k_{i,j}$ denote a partial safari route that is adjustable only on polygon $P_h$. If the only adjustment is a shift adjustment at a vertex $v$ of $P_h$, any subsequent one-place-adjustable route $R^l_{i,j}$ ($l \geqslant k + 2$) in the rest of the process of computing the shortest partial safari route to $v_{i,j}$, cannot be adjustable at $v$ on $P_h$ again. If it is a crossing adjustment, then $P_h$ can never be active again.*

**Proof.** Suppose first that the only adjustment to $R^l_{i,j}$ is a shift adjustment at a vertex $v$ of polygon $P_h$. Since $R^k_{i,j}$ is one-place-adjustable, each part of $R^k_{i,j}$, from $s$ to $v$ and from $v$ to $v_{i,j}$, is the shortest partial safari route. It then follows from Lemma 2 that any subsequent one-place-adjustable route $R^l_{i,j}$ ($l \geqslant k + 2$) in the rest of the process of computing the shortest partial safari route to $v_{i,j}$ cannot have a shift adjustment at vertex $v$ of $P_h$ again.

Consider now the case that the only adjustment is a crossing adjustment at the vertex $v$ of $P_h$. Without loss of generality, we assume that the wrapped contact of route $R^k_{i,j}$ with $P_h$ is only the vertex $v$. Since two parts of route $R^k_{i,j}$, one from $s$ to $v$ and the other from $v$ to $v_{i,j}$, are the shortest, any shorter partial safari route to $v_{i,j}$ cannot reflect on $P_h$ at $v$. Assume that a recovery test makes $P_h$ active, which gives a route $R^l_{i,j}$ with $l \geqslant k+2$. Assume also that $R^l_{i,j}$ reflects at some point $x$, say, to the left of $v$, on the boundary of $P_h$. Since polygon $P_h$ is convex, the part of route $R^l_{i,j}$ from $x$ to $v_{i,j}$ has to first overlap with the boundary of $P_h$ from $x$ to $v$ and then follow the part of $R^k_{i,j}$ from $v$ to $v_{i,j}$. It contradicts the assumption that route $R^l_{i,j}$ reflects on an edge of $P_h$. The proof is completed. □

**Theorem 1.** *The shortest fixed safari route through $s$ can be found in $\mathrm{O}(n^3)$ time, where $n$ is the total number of vertices of $P$ and polygons of $\mathcal{P}$.*

**Proof.** Let us first consider the time required to compute the shortest partial safari route to a vertex $v_{i,j}$ on a polygon $P_i$. It follows from Lemma 4 that once an active polygon becomes inactive because of a crossing adjustment, it can never be active again. Hence, the total number of crossing adjustments performed is less than $i-1$, and so is the total number of the polygons made active by recovery tests. Also, it follows from Lemma 4 that the number of shift adjustments required is $\mathrm{O}(n)$. Thus, the total time taken by adjustments and recovery tests is $\mathrm{O}(n^2)$, which dominates the time spent to compute the shortest partial safari route to $v_{i,j}$.

Since the total number of vertices of polygons in $\mathcal{P}$ is $\mathrm{O}(n)$, the time taken to compute the shortest fixed safari route through $s$ is $\mathrm{O}(n^3)$. □

## 4. Removal of the starting point restriction

The restriction of giving a starting point can be removed by the following brute-force method. Observe that a shortest safari route either reflects at at least one vertex of some polygon in $\mathcal{P}$ (the reflection is usually not perfect), or makes perfect reflections on all active edges. The first case can be handled by taking each polygon vertex as a starting point and thus finding $\mathrm{O}(n)$ shortest fixed safari routes. (Note that when a vertex of polygon $P_i$ is taken as a starting point, the computed route does not allow to enter the interior of $P_i$.) In the second case, any shortest safari route can be considered as the result of an adjustment to some shortest fixed safari route. So such a shortest safari route can be found by performing all possible adjustments to the shortest fixed safari routes. Since no starting point is given, the new problem is how such an adjustment can be performed. Since removing a wrapped polygon (i.e., a crossing adjustment) can be considered as a shift adjustment on its preceding or successive reflected polygon (Fig. 3(a)), it suffices to consider only shift adjustments. Instead of the starting point $s$, we have a starting edge $e$, to which the direction of the shift adjustment points. It is then reduced to the problem of finding the shortest path between edge $e$ and its image in the unfolded polygon, which can be solved in linear time [5]. Hence, a shortest safari route can be found, after $\mathrm{O}(n)$ adjustments to the shortest fixed safari routes are made. In conclusion, we have the following result.

**Theorem 2.** *Let $P$ be a simple polygon, and let $\mathcal{P}$ be a set of disjoint convex polygons inside $P$, each sharing one edge with $P$. A shortest safari route can be found in $\mathrm{O}(n^4)$ time, where $n$ is the total number of vertices of $P$ and polygons of $\mathcal{P}$.*

## Appendix A

In this appendix, we show that the time analysis of Ntafos' safari route algorithm [9] is erroneous and no time bound is guaranteed for his algorithm.

Ntafos' algorithm for computing the shortest fixed safari route is based on the known zookeeper's route algorithms [4,7]. Suppose that $R$ is a shortest zookeeper's route that visits (without entering) every active polygon of $\mathcal{P}'$, which initially equals to $\mathcal{P}$. Clearly, $R$ is a safari route and can be shortened by allowing it to cross the wrapped polygons. Without loss of generality, we assume that the contact of route $R$ with a wrapped polygon is a single vertex and $R$ wraps around only one polygon between successive reflections [9]. Let $R_{\text{target}}$ denote the route obtained by removing all wrapped polygons and computing the shortest zookeeper's route for the new set $\mathcal{P}'$ of active polygons. Note that route $R_{\text{target}}$ may not be a safari one. Define the *apex* of a wrapped polygon $P_i$ in the unfolded route $R$ as its wrapped vertex, and the *dislocation* as the ratio $h_i/\min(x_{i1}, x_{i2})$, where $h_i$ is

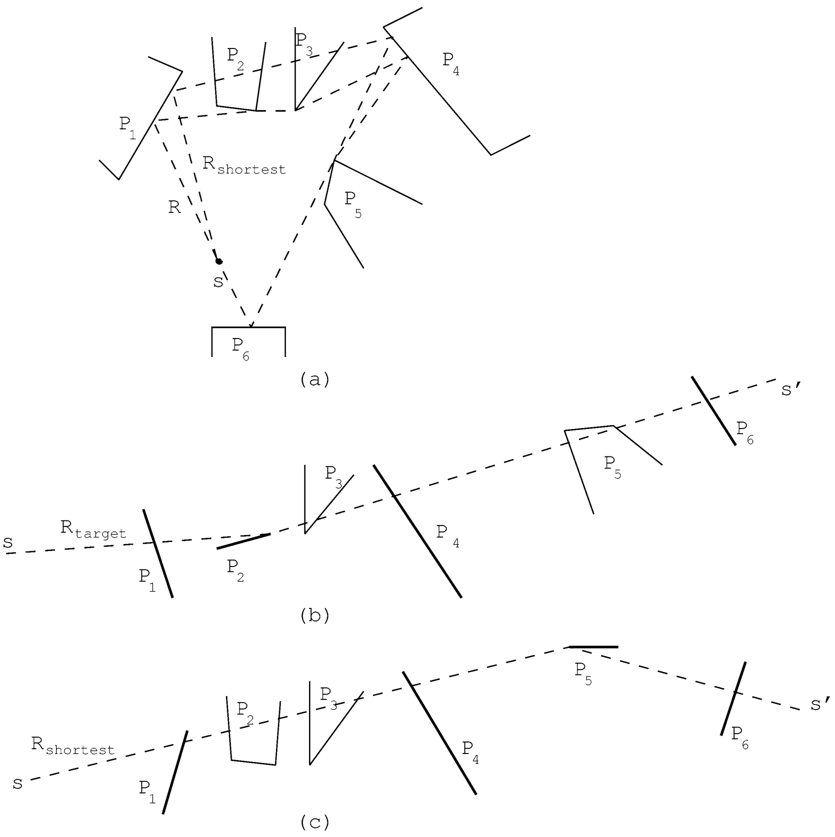

Fig. 6. A counterexample to the maximum dislocation selection rule.

the distance from the apex of $P_i$ to the unfolded route $R_{\text{target}}$ measured along a perpendicular that intersects the unfolded route $R_{\text{target}}$ at $y_i$ and $x_{i1}$, $x_{i2}$ are the distances from $y_i$ to $s$, $s'$ measured along the unfolded route $R_{\text{target}}$, respectively. The dislocation is negative when the wrapped polygon is not crossed by the unfolded route $R_{\text{target}}$. Ntafos claimed that if the maximum (positive) dislocation is due to $P_i$, the shortest fixed safari route crosses $P_i$ (Lemma 5 of [9]). This gives a method to identify a sequence of removals so that the removal of a wrapped polygon early in the sequence need not be undone by later removals. The proof of Ntafos' claim mainly depends on the observation that the unfolded version of the shortest fixed safari route $R_{\text{shortest}}$ is contained in the envelope defined by the unfolded versions of the routes $R_{\text{even}}$ and $R_{\text{odd}}$, which are obtained by removing the wrapped polygons in $R$ with even index and odd index (in our description), respectively.

Our small counterexample to Ntafos' claim is shown in Fig. 6. It suffices to look at three polygons $P_2$, $P_3$ and $P_5$. The initial route $R$ reflects on $P_2$ and wraps around $P_3$, $P_5$, while the route $R_{\text{shortest}}$ reflects on $P_5$ and crosses $P_2$, $P_3$. (The simple polygon $P$ is not important and thus omitted in Fig. 6.) However, if we compute the route $R_{\text{target}}$ by removing $P_3$ and $P_5$ from the set of active polygons in $R$, then the maximum dislocation is due to $P_5$. It follows from Ntafos' claim that the route $R_{\text{shortest}}$ should cross $P_5$, a contradiction.

Let us explain a little more with the aid of unfolded routes. In Fig. 6(b), the unfolded route $R_{\text{target}}$ is given. The reflected polygons are shown only by their active edges, which are drawn in bold lines. (Because of the difficulty of unfolded figures, they are not drawn so precisely, but enough to show where the problem is.) Since the distance from the apex of $P_3$ to $R_{\text{target}}$ can be very small, the dislocation due to $P_5$ is larger than that due to $P_3$. On the other hand, the apex of $P_5$ is contained in the triangle determined by $s$, $s'$ and the apex of $P_2$ in Fig. 6(b). This implies that the route $R_{\text{shortest}}$ should reflect on $P_5$ and cross $P_2$ and $P_3$. See Fig. 6(c). Since both positions of $s$ and $s'$ in Fig. 6(b) cannot be the same as those in Fig. 6(c), the observation that the unfolded route $R_{\text{shortest}}$ is contained in the envelope of the unfolded routes $R_{\text{odd}}$ and $R_{\text{even}}$ was wrong. This is the reason why Ntafos' claim is not generally true. (The same error also occurred in the time analysis of previous watchman route algorithms [3,10]. This error is first pointed out by Hammar and Nillson [8], and finally fixed using dynamic programming by Tan et al. [11].) It seems that the only way to make sure that no adjustments previously made can be undone is to perform an adjustment only when the current route is one-place-adjustable.

## References

[1] B. Chazelle, Triangulating a simple polygon in linear time, Discrete Comput. Geom. 6 (1991) 485–524.

[2] W.P. Chin, S. Ntafos, Optimum watchman routes, Inform. Process. Lett. 28 (1988) 39–44.

[3] W.P. Chin, S. Ntafos, Shortest watchman routes in simple polygons, Discrete Comput. Geom. 6 (1991) 9–31.

[4] W.P. Chin, S. Ntafos, The zookeeper route problem, Inform. Sci. 63 (1992) 245–259.

[5] J. Czyzowicz, P. Egyed, H. Everett, D. Rappaport, T. Shermer, D. Souvaine, G. Toussaint, J. Urrutia, The aquarium keeper's problem, in: Proc. 2nd ACM–SIAM Symp. Discrete Algorithms, 1991, pp. 459–464.

[6] L. Guibas, J. Hershberger, D. Leven, M. Sharir, R. Tarjan, Linear time algorithms for visibility and shortest path problems inside simple triangulated polygons, Algorithmica 2 (1987) 209–233.

[7] J. Hershberger, J. Snoeyink, An efficient solution to the zookeeper's problem, in: Proc. 6th Canadian Conf. on Comput. Geom., 1994, pp. 104–109.

[8] M. Hammar, B.J. Nilsson, Concerning the time bounds of existing shortest watchman route algorithms, in: Proc. FCT'97, in: Lecture Notes in Comput. Sci., Vol. 1279, Springer, Berlin, 1997, pp. 210–221.

[9] S. Ntafos, Watchman routes under limited visibility, Comput. Geom. Theory Appl. 1 (1992) 149–170.

[10] X. Tan, T. Hirata, Y. Inagaki, An incremental algorithm for constructing shortest watchman routes, Internat. J. Comput. Geom. Appl. 3 (1993) 351–365.

[11] X. Tan, T. Hirata, Y. Inagaki, Corrigendum to "An incremental algorithm for constructing shortest watchman routes", Internat. J. Comput. Geom. Appl. 9 (1999) 319–323.

[12] X. Tan, Fast computation of shortest watchman routes in simple polygons, Inform. Process. Lett. 77 (2001) 27–33.