Touring a Sequence of Line Segments in Polygonal Domain Fences

Amirhossein Mozafari* mozafari@alum.sharif.edu
Alireza Zarei* zarei@sharif.edu

Abstract

In this paper, we consider the problem of touring a sequence of line segments in presence of polygonal domain fences. In this problem there is a sequence $\mathcal{S} = (s = S_0, \ldots, S_k, S_{k+1} = t)$ in which s and t are respectively start and target points and $S_1, \ldots, S_k$ are line segments in the plane. Also, we are given a sequence $\mathcal{F} = (F_0, \ldots, F_k)$ of planar polygonal domains called fences such that $S_i \cup S_{i+1} \subset F_i$. The goal is to obtain a shortest path from s to t which visits in order each of the segments in $\mathcal{S}$ in such a way that the portion of the path from S_i to S_{i+1} lies in F_i. In 2003, Dror *et. al.* proposed a polynomial time algorithm for this problem when the fences are simple polygons. Here, we propose an efficient polynomial time algorithm for this problem when the fences are polygonal domains (simple polygons with some polygonal holes inside).

1 Introduction

Computing a shortest path between two points having some desired properties is one of the classic and well studied problems in computer science and computational geometry. In many applications, we need to visit a sequence of certain regions while traversing from s to t. We call this class of problems as visiting problems. In such visiting problems, a desired path may be restricted to a fence (or fences) which means that the path must completely lie inside a special region. Zoo-Keeper [7], Safari [11] and Watchman Route [4] problems are famous examples of such visiting problems. In Zoo-keeper and Safari problems we need to obtain a shortest tour visiting a set of disjoint convex polygons (called cages) inside a simple polygon P (the fence) each of which shared an edge with the boundary of P. The difference between these two problems is that in the first one, the desired path cannot enter into the cages while this restriction does not exist in the second one. In watchman route problem (fixed source version) we have a point inside a simple polygon P (the fence) and we seek for a shortest tour inside P containing s such that every point in P is visible from at least one point of the tour.

*Department of Mathematical Science, Sharif University of Technology

Dror *et. al.* [5] in 2003 introduced a general version of these visiting problems called *touring polygons problem (TPP)*. In this problem there is a sequence $\mathcal{P} = (s = P_0, \ldots, P_k, P_{k+1} = t)$ of polygons where s and t are respectively start and target points and a sequence $(F_0, \ldots, F_k)$ of simple polygonal fences where $P_i \cup P_{i+1} \subset F_i$. The goal of this problem is to obtain a shortest path from s to t which intersects the polygons of $\mathcal{P}$ in order and its subpath from P_i to P_{i+1} lies inside F_i. They proved that TPP is NP-hard for intersecting polygons and proposed a $O(nk^2 \log n)$ time algorithm for convex polygons. They also gave a $O(kn \log(n/k))$ time algorithm for the cases where the polygons are convex and pairwise disjoint and the fences are the whole plane. Here, n is the total number of vertices of all fences and polygons. In 2006, Arkin *et al.* [3] considered the touring polygons problem in L_1 metric for the cases where polygons are pairwise disjoint segments and the fences are the whole plane and proposed a $O(k^2)$ time algorithm for this version.

For one decade the complexity of TPP for disjoint non-convex polygons was unknown and during these years several approximation algorithms have been proposed for solving this version of the problem [6, 10]. Finally, Ahadi *et al.* [1] in 2013 proved that TPP is NP-hard for disjoint polygons in any L_p norm.

Despite many investigations and results on various kinds of visiting problems with simple polygon fences, there are less results on visiting problems whose fences are polygonal domains. In 2014, Ahadi *et al.* [2] gave a polynomial time algorithm for touring polygonal objects problem in which the fences are polygonal domains. This problem is similar to TPP but in this problem a desired path cannot enter into the polygons (this is like the Zoo-Keeper problem in which the tour cannot enter the cages).

In this paper, we present a polynomial time algorithm for a version of the TPP called *Touring Line Segments Problem (TLSP)* in which the polygons are line segments and the fences are polygonal domains. This is the first polynomial time algorithm which considers TPP when the fences are polygonal domains. Figure 1 shows an example of TLSP. We show that this problem can be solved in $O(n^3k)$ time where n is the number of vertices of all fences and segments and k is the number of segments. We use the well-known continuous Dijkstra paradigm [9, 8] to obtain a shortest path for our

problem.

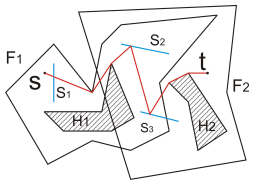

Figure 1: An example of TLSP where (s, S_1, S_2, S_3, t) is the sequence of segments and (PLN, F_1, PLN, F_2) is the sequence of fences and PLN represents the whole plane.

2 Preliminaries and Definitions

Let $\mathcal{T} = (\mathcal{S}, \mathcal{F})$ be an instance of TLSP with segments $\mathcal{S} = (s = S_0, S_1, \ldots, S_k, S_{k+1} = t)$ and fences $\mathcal{F} = (F_0, \ldots, F_k)$ in the plane such that $S_i \cup S_{i+1} \subset F_i$.

Observation 1. Any optimal path is a polygonal chain which bends only on some vertices of the fences, endpoints of segments, intersection point of two consecutive segments or on some reflection points on the interior of segments of $\mathcal{S}$.

According to the above observation, when an optimal path intersects the interior of a segment (not on the sgment endpoints) according to its order, the optimal path either passes or reflects on this segment. We denote by $\mathcal{T}_i(x)$ an instance of TLSP with $(S_0, \ldots, S_i, x)$ and $(F_0, \ldots, F_i)$ as its segments and fences, respectively. The set of optimal paths of $\mathcal{T}_i(x)$ is denoted by $P_i(x)$. Let $x \in F_i$ and $p \in P_i(x)$ and v be a vertex of F_j or an endpoint of S_j $(j \leq i)$. We call v an *origin* of x if v is the last fence vertex or segment endpoint on p when we traverse p from s to x. Thus, p must pass or reflect on the remaining segments from S_{j+1} to S_i to reach x. We say that v' is a *source* of x if it is obtained by the sequence of reflections of v on the supporting lines of segments that p reflects from v to x in order. For example, s is the *origin* of points x,x',x'' and x''' in Figure 2, and s,s',s'' and s''' are respectively the corresponding sources of the points x,x',x'' and x'''. According to this definition, the length of the portion of p from v to x is equal to $|v'x|$. According this definition, the number of distinct sources of all points in F_i can exponentially grows. Figure 2 shows an example of such situations. In this figure, if we consider $\mathcal{S} = (s)$, s is the only source of the points in the plane. After adding S_1 to $\mathcal{S}$, the set of sources of points in the plane becomes $\{s, s'\}$ and after adding S_3, the sources are $\{s, s', s'', s'''\}$. In this example, each segment doubles the set of sources of all points of the plane. We say that a point $x \in F_i$ is *straightly reachable from* $S_j (1 \leq j \leq i)$ if it has an optimal path with an origin in vertices of F_l or an endpoint of S_l where $l < j$.

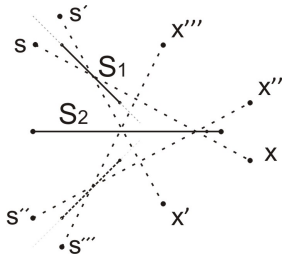

Figure 2: The number of sources can grow exponentially: s,s',s'' and s''' are respectively the corresponding sources of the points x,x',x'' and x'''.

To find an optimal path for $\mathcal{T}$, we use the continuous Dijkstra paradigm [9, 8] (In Appendix 1, we briefly describe the continuous Dijkstra paradigm using our terminology). Basically, we need some modifications on this paradigm to enforce shortest paths to visit the segments in order from s to t. But, as we said before, the number of sources of all points of the fences may grow exponentially and we cannot consider them as the set of initial sites in the continuous Dijkstra paradigm to obtain a shortest path map for each fence from which an optimal path is obtained. In next sections, we first prove some properties about optimal paths and based on these properties a modified version of the continuous Dijkstra paradigm is proposed to solve TLSP in polynomial time.

3 TLSP for Consecutively Disjoint Segments

In this section, we restrict ourselves to instances of TLSP in which the segments of $\mathcal{S}$ are consecutively disjoint. Our algorithm can be extended to solve TLSP for intersecting segments as described in Appendix 4 and we skip it because of the limited space here. Let $\mathcal{T} = (\mathcal{S}, \mathcal{F})$ be such an instance of the problem. If S_i, for $1 \leq i \leq k$, is a single point, we can break this problem into two sub-problems one from s to S_i and another from S_i to t. Therefore, we assume that s and t are the only points in $\mathcal{S}$. To obtain an optimal path for $\mathcal{T}$, we use the continuous Dijkstra paradigm to build a shortest path map for each F_i $(0 \leq i \leq k)$ namely SPM_i such that having the sequence $\mathcal{SPM} = (SPM_0, \ldots, SPM_k)$ we can obtain a solution for $\mathcal{T}$. For simplicity, we imagine $k+1$ planes such that the i^{th}-plane $(0 \leq i \leq k)$ contains only S_i, F_i and S_{i+1}, and construct SPM_i in this plane. To construct the shortest path maps we need $k+1$ wavefronts each of which propagates in one plane and sweeps its fence. To identify the initial wavelets of these wafefronts, we need some information about the configuration of $\mathcal{T}$. For this purpose, our algorithm consists of three phases: pre-processing $\mathcal{T}$, building $\mathcal{SPM}$

and computing an optimal path.

3.1 The pre-processing phase

Before we describe the first phase, we need some definitions. We inductively define the extensions of S_j in i^{th}-plane ($0 < j \leq i$) as follows: For $i = j$, the extensions of S_j is simply the two half-lines along $\bar{S}_j$ starting from its endpoints and going away from S_j where $\bar{S}_j$ is the supporting line of S_j. For $i > j$, let r be the intersection of S_i and an extension e of S_j in the $(i-1)^{th}$-plane. For any one of these intersection points, the half-line from r along e and its reflection with respect to $\bar{S}_i$ are extensions of S_j in the i^{th}-plane. In fact, each extension of S_j in the $(i-1)^{th}$-plane which intersects S_i, generates two extensions of S_j in the i^{th}-plane (If S_i does not intersect any extension of S_j in the $(i-1)^{th}$-plane, we have no extension of S_j in the i^{th}-plane)(See Figure 3). According to this definition, S_i and extensions of S_j in the i^{th}-plane induces a subdivision which is called the j^{th}-subdivision of the i^{th}-plane.

Lets assign α and β marks arbitrary to the sides of the supporting line of each segment $S_j \in \{S_1, \ldots, S_k\}$. Based on this *initial marking assignment*, we mark all regions of the j^{th}-subdivision of the i^{th}-plane ($i \geq j$) as follows: For $i = j$, we mark the half-plane lies on α-side of S_j as α-region and the other as β-region. For $i > j$, each region R of the j^{th}-subdivision of the i^{th}-plane contains exactly one sub-segment of S_i on its boundary. We assign to R the mark of the region of the j^{th}-subdivision of the $(i-1)^{th}$-plane which contains this sub-segment. Note that according to our definition, this sub-segment must entirely lie in one region of the j^{th}-subdivision of the $(i-1)^{th}$-plane (Figure 3).

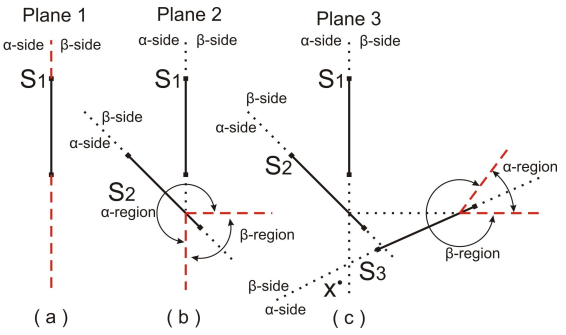

Figure 3: Extensions of S_1 in the 1^{th},2^{th} and 3^{th}-plane are shown by dashed half-lines.

Let x be a point in the i^{th}-plane. The j^{th}-component of x for $1 \leq j \leq i$ is defined to be the mark (α or β) of that region of the j^{th}-subdivision of the i^{th}-plane which contains x. According to Observation 1 and this definition, it is simple to verify that if x is straightly reachable from S_j ($1 \leq j \leq i$) in some path $p \in P_i(x)$, then, for all $j \leq l < i$, the mark of the region of the j^{th}-subdivision of the l^{th}-plane in which p traverses from S_l to S_{l+1} is equal to the j^{th}-component of x. This fact gives us an intuition about the optimal paths to x that their origins are vertices of $F_l \cup S_l$ for some $l < j$. These paths will reflect on or pass through segments $S_j, S_{j+1}, \ldots, S_i$ and their traversal (passing through or reflecting on) from S_j to S_i is according to the j^{th}, $j+1^{th}$, $\ldots$,i^{th} component of x.

Let $\mathcal{M}$ be an initial marking assignment for sides of the supporting lines of segments $\{S_1, \ldots, S_k\}$. The characteristic sequence of a point $x \in i^{th}$-plane is defined as a vector $< x_1, \ldots, x_i >$ in which its j^{th}-term is computed as follows (this has been described formally in Algorithm 1 in Appendix 3. In this algorithm, procedure PChar(j,i,x) is used to find the j^{th}-term of the characteristic sequence of a point $x \in i^{th}$-plane). The last term x_i is equal to the mark of that side of $\bar{S}_i$ which contains x. For $j < i$, the j^{th}-term is recursively defined to be either the j^{th}-term of the characteristic sequence of x in the $(i-1)^{th}$-plane or the j^{th}-term of the characteristic sequence of the reflection of x on $\bar{S}_i$ in the $(i-1)^{th}$-plane. The former case happens when x lies in the α side of $\bar{S}_i$ and the later is used otherwise.

Note that the j^{th}-term of the characteristic sequence of a point in the i^{th}-plane ($i > j$) is not necessarily equal to its j^{th}-component. For example, the 1^{th}-component of $x \in 3^{th}$-plane in Figure 3 is β while its first characteristic sequence term is α. However this difference is due to the initial marking assignment used in this example.

Assume that e is an extension of S_j in the $(i-1)^{th}$-plane ($j < i$) which starts from point r on S_{i-1} and intersects S_i. We say that an initial marking assignment $\mathcal{M}$ has *passing property* if for all such extensions, r lies in β-side of S_i. The following lemma implies that there is always an initial marking assignment of $\mathcal{S}$ having passing property.

Lemma 1. The starting point of all extensions of all segments $S_j \in \{S_1, \ldots, S_{i-1}\}$ in the $(i-1)^{th}$-plane that intersect S_i lie on one side of $\bar{S}_i$.

Proof. See Appendix 2.□

Lemma 1 implies that there is always an initial marking assignment of sides of segments $\{S_1, \ldots, S_k\}$ having passing property and shows how to obtain such an assignment. But, before describing a method for obtaining such a marking assignment we describe how such a marking is used in obtaining optimal paths. As said before, identifying all j^{th}-component of a point $x \in i^{th}$-plane ($1 \leq j \leq i$) are critical in obtaining an optimal path to x. The following lemma shows how we can obtain these data.

Lemma 2. Let $\mathcal{M}$ be an initial marking assignment of segments which satisfies the passing property, and $x \in F_i$ is straightly reachable from S_j ($j \leq i$). Then, the j^{th}-term of the characteristic sequence of this point

for marking $\mathcal{M}$ is equal to its j^{th}-component.

Proof. See Appendix 2.□

Now, we return to the concept of the initial marking assignment and propose a method for obtaining a marking which has passing assignment. This method has been formally described as procedure MarkSides($\mathcal{S}$) in Algorithm 2 in Appendix 3.

Our marking algorithm iteratively marks both sides of segments from S_1 to S_k. According to the definition of the passing property, the starting points of all extensions of $S_1, \ldots, S_{i-1}$ that intersect S_i in the $(i-1)^{th}$-plane lie in the β-side of $\bar{S}_i$. Also, from the definition of the extensions in the $(i-1)^{th}$-plane, all these starting points lie on S_{i-1}. For S_1, we do not have any preceding segment and $s = S_0$ is a point which can be considered as a degenerate segment. Therefore, we mark the containing half-plane of s as β-side of $\bar{S}_1$ and the other as α-side. Then, each half-line extension of S_0 that starts from s and intersects S_1 has its starting point in the β-side of S_1. Therefore, marking S_1 this way supports the passing property.

Now assume that we have marked sides of segments $S_1, \ldots, S_{i-1}$ in such a way that supports the passing property. If S_{i-1} lies completely in one side of $\bar{S}_i$, the two extensions of S_{i-1} (from its endpoints) as well as extensions of preceding segments that intersect S_{i-1} have starting point on one side of $\bar{S}_i$. To support the passing property, we mark this side as β-side and the other as α-side. Otherwise (if S_{i-1} intersects $\bar{S}_i$), S_i must lie completely in one side of $\bar{S}_{i-1}$. This is due to our assumption that segments are consecutively disjoint. Then, none of the two extensions of S_{i-1} intersects S_i. Therefore, the extensions that intersect S_i in the i^{th}-plane are either extensions in $(i-2)^{th}$-plane that intersect S_{i-1} or the reflections of these extensions. Thinking inductively, we have already marked the sides of $\bar{S}_{i-1}$ which supports the passing property. This implies that all of the extensions of the $(i-2)^{th}$-plane that intersect S_{i-1} continue in α-side of $\bar{S}_{i-1}$ and their reflections continue in β-side. Then, if S_i lies in α-side of $\bar{S}_{i-1}$ it can be intersected by extensions of the $(i-2)^{th}$-plane and if it lies in β-side, it may be intersected by reflections of these extensions on S_{i-1}. For the first case, all extensions that intersect S_i have a starting point on S_{i-2}. This fact helps us to ignore S_{i-1} and mark sides of S_i considering S_{i-2} as its preceding segment. If we do this inductively, we will finally reach a base case from which the marks of sides of S_i are obtained. In the other case where S_i lies in β side of S_{i-1}, we first reflect S_i on $\bar{S}_{i-1}$ and obtain the marks for sides of this new segment namely S'_i. Then, the marks of the sides of S_i will be the marks of the corresponding sides of S'_i. It is important to note that the extensions of the $(i-1)^{th}$-plane diverse in both α and β sides of $\bar{S}_{i-1}$. This forces that when $\bar{S}_i$ intersects S_{i-1} it will be intersected by only extensions whose starting points from S_{i-1} lie on one side of $\bar{S}_i$.

There is still one flaw in this inductive algorithm. We assumed that consecutive segments do not intersect each other. But, when we reflect S_i on $\bar{S}_{i-1}$ and consider S_{i-2} as the preceding segment of S_i (ignoring S_{i-1}), S_i may intersect S_{i-2}. To solve this problem, we cut that part of S_{i-2} which lies on the α-side of S_{i-1}. Precisely, in each iteration after marking sides of a segment S_i we cut and remove from S_{i-1} that part that lies in α-side of S_i (line 16 in Algorithm 2 in Appendix 3). This does not affect our algorithm because non of the extensions of the $(i-2)^{th}$-plane that have their starting point on the α-side of $\bar{S}_{i-1}$ intersect S_{i-1}, and consequently, does not affect the passing property of S_i.

According to the above discussion, the pre-processing phase of our algorithm consists of two steps. First, marking the sides of $\mathcal{S}$ using procedure MarkSides and second, computing the characteristic sequences of all vertices of F_i and endpoints of S_{i+1} in the i^{th}-plane $(1 \leq i \leq k)$. These information enables us to efficiently construct the shortest path maps in the second phase of our algorithm.

3.2 Building shortest path maps

In order to build the $k+1$ shortest path maps $\mathcal{SPM} = (SPM_0, \ldots, SPM_k)$, we need to perform in parallel $k+1$ instances of our modified version of the continuous Dijkstra algorithm inside fences $F_0, \ldots, F_k$ in their corresponding planes. The initial wavefront of each F_i propagates from S_i and sweeps F_i to build SPM_i. For F_0, the initial wavefront is a complete circle with zero radius around s. For $i > 0$, the initial wavefront of F_i is determined according to the wavelets intersect S_i during the wavelet propagation of the wavefront of F_{i-1}. Precisely, when a wavelet ω in F_{i-1} intersects an endpoint of S_i, a new wavelet centred at this endpoint and zero radius starts to propagate in F_i. If ω intersects the interior of S_i, it can generate one or two wavelets in F_i according to the information we obtained in the pre-processing phase. In order to detect the wavelets which intersect S_i in F_{i-1} in our implementation of the continuous Dijkstra paradigm, we consider S_i as an obstacle (hole) in F_{i-1}. So, we can use the standard event handling of continuous Dijkstra paradigm [9, 8] to identify such events. Let ω be a wavelet in F_{i-1}. When it touches the interior of S_i, it can generate two wavelets in F_i. One, called *passing wavelet*, propagates in F_i along ω and the other, called *reflecting wavelet*, is the reflection of the passing wavelet with respect to $\bar{S}_i$. Let ω be a wavelet in F_{i-1} with center C_ω which intersects $S_i = ab$ at point I. Without lose of generality, assume that C_ω lies in the α-side of S_i. Two cases may happen: First, I is a point in the interior of ω and second, I is an endpoint of ω. In the first case, the passing wavelet of ω is a wavelet with center C_ω which propagates from I in

the α-side of S_i and is restricted to the angle $\widehat{aC_\omega b}$ in F_i (See Figure 4).

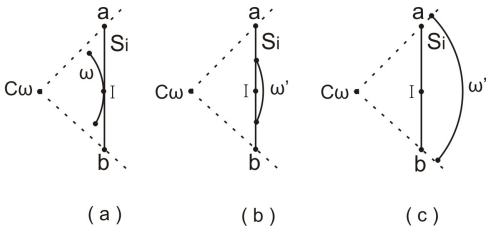

Figure 4: ω generates a passing wavelet ω' in F_i when an interior point of ω intersects S_i.

In the second case, the passing wavelet of ω is a wavelet with center C_ω which propagates from I in the α-side of S_i. This wavelet is restricted to the angle $\widehat{aC_\omega I}$ if a lies on that side of $C_\omega I$ which contains ω, and is restricted to $\widehat{IC_\omega b}$, otherwise (Figure 5).

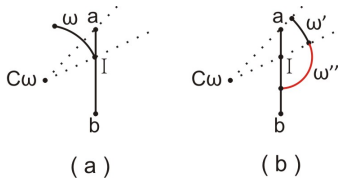

Figure 5: ω generates a passing wavelet ω' in F_i when an endpoint of ω intersects S_i. In this case, another wavelet (ω'') from this intersection point is propagated in F_i.

Note that the center of passing and reflecting wavelets may lie outside F_i. According to the above discussion, the initial wavelets of F_i are the wavelets propagate from the endpoints of S_i (these endpoints act like initial sites in F_i whose weights are assigned by the wavefront of F_{i-1}) and a set of passing or reflecting wavelets obtained according to the pre-processing information. Moreover, when a wavelet ω in F_{i-1} generates a passing or reflecting wavelet in F_i by intersecting S_i in its endpoint (the second case in the above discussion) we consider the intersection point (I in Figure 5) as an initial site in F_i whose weight is assigned by ω in F_{i-1}.

If we consider both passing and reflecting wavelets for all wavelets that intersect S_i in F_{i-1} for all $1 \leq i \leq k+1$, the corresponding cells of passing (resp. reflecting) wavelets in F_i represent the set of points $x \in F_i$ for which there is an optimal path in $P_i(x)$ that passes through (resp. reflects on) S_i. Furthermore, the site of each cell (center of the wavelet which sweeps this cell) in F_i will be a source of its containing points and for each $x \in F_i$ we can recursively construct an optimal path in the same way as what we do in obtaining an optimal path between two points in a polygonal domain using continuous Dijkstra paradigm [9]. But, as we said in Section 2, the number of sources and therefore the number of cells can grow exponentially this way. To overcome this problem, we filter the passing and reflecting wavelets generated by the wavelets in F_{i-1} according to the pre-processing information. For this purpose, we assign a sequence $SQ(\omega)$ to each wavelet ω in F_i $(0 \leq i \leq k)$ as follows: The sequence of all wavelets in F_i whose center is a vertex of F_i or an endpoint of S_i is the empty sequence. If a wavelet ω in F_{i-1} generates a wavelet ω' in F_i (by passing or reflecting) which propagates in α-side (resp. β-side) of S_i, we have $SQ(\omega') = (SQ(\omega), \alpha)$ (resp. $SQ(\omega') = (SQ(\omega), \beta)$).

Let $\mathcal{P}_i$ be the set of all vertices of F_i and endpoints of S_{i+1} $(1 \leq i \leq k)$ and ω be a wavelet in F_{i-1} with $SQ(\omega) = (w_1, \ldots, w_l)$ which intersects the interior of S_i. The wavelet ω generates a wavelet in F_i only when there exists a point $x \in \mathcal{P}_j$ $(j \geq i)$ with characteristic sequence $(x_1, \ldots, x_{i-1}, x_i, \ldots, x_j)$ where $(w_1, \ldots, w_l) = (x_{i-l}, \ldots, x_{i-1})$. Then, if $x_i = \alpha$, ω generates a wavelet (by passing or reflecting) in α-side of S_i and if $x_i = \beta$, ω generates a wavelet in β-side of S_i in F_i. This filtering mechanism prevents the exponential growth of the wavelets in our algorithm.

Lemma 3. The total size of the shortest path maps $\mathcal{SPM}$ obtained according to the above propagation rules is polynomial in terms of the number of vertices of all fences and k.

Proof. The complexity of $SPM_i \in \mathcal{SPM}$ is proportional to the number of wavelets propagates in F_i. Trivially, this is linear with respect to the number of vertices in F_0 for SPM_0. The wavelets of SPM_i for $i > 0$ are generated due to either a vertex in F_i or intersecting S_i by a wavelet in SPM_{i-1}. For the first case, each vertex of F_i generates exactly one wavelet. Wavelets of the second type have non-empty sequences. For each wavelet with non-empty sequence $(w_1, \ldots, w_l)$ in F_i, there must exist a vertex in $\mathcal{P}_j$ with characteristic sequence $(x_1, \ldots, x_{i-1}, x_i, \ldots, x_j)$ where $(w_1, \ldots, w_l) = (x_{i-l+1}, \ldots, x_i)$. Therefore, the number of distinct wavelet sequences of length l is proportional to the number of vertices in $\bigcup \mathcal{P}_j$. On the other hand, all wavelets with sequence of length l in F_i have emerged in a vertex of F_{i-l}. Therefore, $|\mathcal{P}_{i-l}|.|\bigcup_{j=k}^{i} \mathcal{P}_j|$ is an upper bound for the number of wavelets in F_i with sequence of length l. Summing this for all $1 \leq l \leq i$ results the lemma.□

Lemma 4. The weights assigned by the wavefront in SPM_i to each vertex v of $F_i \cup S_{i+1}$ is equal to the length of all optimal paths for $\mathcal{T}_i(v)$.

Proof. We prove this lemma by induction on i. According to the standard continuous Dijkstra paradigm, the theorem is true for vertices of $F_0 \cup S_1$. For $i > 0$, let x be a vertex of F_i or an endpoint of S_{i+1}. The vertex x

either has an origin v in the vertices of $F_j \cup S_{j+1}$ where $j < i$ or all of its origins belong to F_i. The first case means that x is straightly reachable from S_{j+1}. According to the induction hypothesis, the weight of v is the optimal length from s to v. Also, by Lemma 2 and the propagation rules, the wavelet with center v in F_j generates wavelets in fences $F_l (j < l \leq i)$ according to the characteristic sequence of x. This means that we traverse along an optimal path from v to x which implies the lemma for x. For the other case where in all optimal paths the origin of x belongs to F_i, let $v \in F_i$ be the origin of x in an optimal path. If v is straightly reachable from S_{i+1} then we have the optimal length to v according to the first case and our version of continuous Dijkstra paradigm computes the shortest path from v to x. Otherwise, we can repeat this process inductively until reaching an endpoint of S_{i+1} or a straightly reachable vertex of F_i from S_{i+1}. □

3.3 Obtaining an optimal path

As the final phase of our algorithm we find an optimal path from s to t. This is a straightforward usage of the data in $\mathcal{SPM}$. For this purpose, we use procedure $OPT(i, x)$ (Described formally in Algorithm 3 in Appendix 3) which returns an optimal path for $\mathcal{T}_i(x)$ as a list of points. Then, $OPT(k, t)$ will be an optimal path for $\mathcal{T}$. In this procedure, if $s = x$, the optimal path for $T_0(s)$ is trivially the single point s. If x is an endpoint of S_i, an optimal path for $\mathcal{T}_i(x)$ is just an optimal path for $\mathcal{T}_{i-1}(x)$ which is the optimal path computed by $OPT(i-1, x)$. Otherwise, there are two cases for the last segment of an optimal path to x in $\mathcal{T}_i(x)$: First, the site c of the cell C in SPM_i which contains x is a vertex of F_i or an endpoint of S_i. Second, this cell belongs to a passing or reflecting wavelet ω. For the first case the optimal path is $(OPT(i, c), x)$ which means that the last segment of the optimal path is cx. For the second case, let v be an origin of x. We obtain a sub-path from v to x by a list L of points according to $SQ(\omega)$. Then, $(OPT(i-|SQ(\omega)|, v), L)$ is an optimal path from s to x which means that the optimal path goes to v and then, from v to x it acts (passes or reflects) according to $SQ(\omega)$.

3.4 Complexity of the Algorithm

In the first phase of our algorithm we compute characteristic sequences of all vertices of fences and endpoints of segments in $\mathcal{T}$. We use procedure MarkSides to properly mark the sides of each segment in $\mathcal{S}$. The function MARK in this procedure takes $O(k)$ time. Therefore, procedure MarkSides can be run in $O(k^2)$ time. Each execution of procedure PChar takes $O(k)$ and each characteristic sequence have $O(k)$ terms which means that obtaining characteristic sequence of each point takes $O(k^2)$ time. Therefore, the time complexity of the first phase of our algorithm is $O(k^2 n)$ where n is the total number of all fences vertices and segments endpoints. In the second phase, as discussed in the proof of Lemma 3, the total number of wavelets in all planes is $O(n^2)$ and we should handle $O(n^2)$ events according to [8]. However, in each event handling procedure, we have to perform $O(k)$ comparison for all $O(n)$ points to decide whether this wavelet must generate a passing or reflecting wavelet in the next plane. Therefore, the complexity of the second phase is $O(n^3 k)$. The complexity of the third phase depends on the number of bends and intersection points on the obtained optimal path which is $O(n)$. We can store enough information in shortest path maps to restore such an optimal path in linear time in terms of the output path length (storing the site of each cell and the corresponding sequence of its wavelet). Summing all these costs implies that the complexity of our algorithm is $O(n^3 k)$.

References

[1] A. Ahadi, A. Mozafari, and A. Zarei. "Touring disjoint polygons problem is NP-hard." In *Combinatorial Optimization and Applications*, pp. 351-360. Springer International Publishing, 2013.

[2] A. Ahadi, A. Mozafari, and A. Zarei. "Touring a sequence of disjoint polygons: Complexity and extension." *Theoretical Computer Science* 556 (2014): 45-54.

[3] E. Arkin, A. Efrat, C. Erten, F. Hurtado, J. Mitchell, V. Polishchuk, and C. Wenk. "Shortest tour of a sequence of disjoint segments in L1." In *Proc. 16th Fall Workshop on Computational and Combinatorial Geometry*. 2006.

[4] W. Chin, and S. Ntafos. "Shortest watchman routes in simple polygons." *Discrete and Computational Geometry* 6, no. 1 (1991): 9-31.

[5] M. Dror, A. Efrat, A. Lubiw, and J. Mitchell. "Touring a sequence of polygons." In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pp. 473-482. ACM, 2003.

[6] F. Li, and R. Klette. "Rubberband algorithms for solving various 2D or 3D shortest path problems." In *Computing: Theory and Applications*, 2007. ICCTA'07. International Conference on, pp. 9-19. IEEE, 2007.

[7] J. Hershberger, and J. Snoeyink. "An Efficient Solution to the Zookeeper's Problem." In *CCCG*, pp. 104-109. 1994.

[8] J. Hershberger, and S. Suri. "An optimal algorithm for Euclidean shortest paths in the plane." *SIAM Journal on Computing* 28, no. 6 (1999): 2215-2256.

[9] J. Mitchell. "Shortest paths among obstacles in the plane." *International Journal of Computational Geometry and Applications* 6, no. 03 (1996): 309-332.

[10] X. Pan, F. Li, and R. Klette. "Approximate shortest path algorithms for sequences of pairwise disjoint simple polygons." In *CCCG*, pp. 175-178. 2010.

[11] X. Tan, and T. Hirata. "Shortest safari routes in simple polygons." In *Algorithms and Computation*, pp. 523-530. Springer Berlin Heidelberg, 1994.

Appendix 1: An Overview on the continuous Dijkstra paradigm

The Dijkstra paradigm is originally proposed to obtain a shortest path between two points namely p and q in a polygonal domain namely D [9, 8]. This is done by simulating the propagation of a wavefront starting to propagate from p and sweeping the entire polygonal domain. Precisely, If we define the length of a shortest path between p and x in D as the weight of x and denote it by $w(x)$, the wavefront at distance d is :

$$WF(d) := \{x \in D | w(x) = d\}$$

A wavefront consists of a set of wavelets each of which is a circular arc whose center is p or some vertex of D. The initial wavefront contains a single point p as its only wavelet (a circle of radius zero centred at p). When a wavefront propagates, its structure (the set of its wavelets) changes, i.e., some wavelets may disappear, some may break into two wavelets and new wavelets may appear. A wavelet is eliminated from the structure of a wavefront when its two neighbour wavelets collide on each other and a wavelet may break into two wavelets when it collides the interior of an edge of D. Also, if a wavefront collides a vertex v of D, a new wavelet with center v appears and starts to propagate in the region of D that v blocks the wavefront to sweep it.

When the wavefront propagates, the traces of these endpoints decompose the swept part of D into regions having this property that all points of a region have combinatorially equivalent shortest paths. Therefore, this wavefront propagation induces a subdivision called *shortest path map* on D (See Figure 6). The *site* of each *cell* of this subdivision is the center of the wavelet that has swept it. If q belongs to a cell with site r, then the last segment of a shortest path from p to q is rq and the length of the shortest path from p to q is $w(r) + |rq|$. We can replace q by r and use the shortest path map to obtain the last segment of a shortest path from p to r and repeat this procedure until a shortest path from p to q is obtained.

We can use the continuous Dijkstra paradigm to solve a more general shortest path problem in which instead of having one initial site (p in the above discussion) we have multiple initial weighted sites namely $\{p_1, \ldots, p_i\}$ with weights $w(p_j)$ $(1 \leq j \leq i)$. Then, for a query point q in D we seek a shortest path from q to an initial site p_j such that $w(p_j) + d(p_j, q)$ is minimum where $d(p_j, q)$ is the length of a shortest path from p_j to q. This can be done by delaying wavelet propagation of each site according to its weight. Hershberger and Suri in [8] gave an implementation of the continuous Dijkstra paradigm which handles such cases.

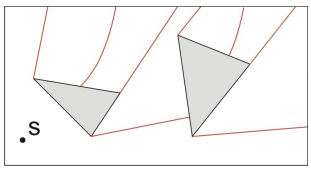

Figure 6: An example of a shortest path map

Appendix 2: Proofs

Lemma 1. The starting point of all extensions of all segments $S_j \in \{S_1, \ldots, S_{i-1}\}$ in the $(i-1)^{th}$-plane that intersect S_i lie on one side of $\bar{S}_i$.

Proof. Let e_1 be an extension of S_{l_1} and e_2 be an extension of S_{l_2} in the $(i-1)^{th}$-plane with starting points r_1 and r_2 respectively which intersect S_i $(1 \leq l_1, l_2 < i)$. We prove that both, r_1 and r_2 lie in one side of $\bar{S}_i$. According to the fact that the definition of extensions is independent to the fences, we assume that the fences are the whole plane. It is simple to see that if x lies on an extension e in the $(i-1)^{th}$-plane, there is always an optimal path $p \in P_{i-1}(x)$ for which e overlaps the last segment of p. For the sake of a contradiction, assume that r_1 and r_2 lie on different sides of S_i. Then, e_1 and e_2 must intersect each other on a point like x. Without loss of generality, we assume that e_1 intersects S_i after x (Figure 7). Then, there will be two optimal paths in $P_{i-1}(x)$: one reaches x from r_1 and the other from r_2. But, this implies that for each point x' after S_i on e_1 in the i^{th}-plane there exist an optimal path with two last segments r_2x and xx' which contradicts Observation 1. □

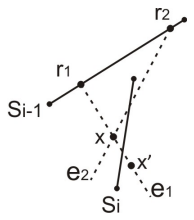

Figure 7: r_1 and r_2 must lie on one side of S_i.

Lemma 2. Let $\mathcal{M}$ be an initial marking assignment of segments which satisfies the passing property, and $x \in F_i$ is straightly reachable from S_j $(j \leq i)$. Then, the j^{th}-term of the characteristic sequence of this point for marking $\mathcal{M}$ is equal to its j^{th}-component.

Proof. Let $x \in F_i$ be straightly reachable from S_j in some optimal path $p \in P_i(x)$. This means that x is straightly reachable from all $S_l (j \leq l \leq i)$. Trivially, all fences $F_j, \ldots, F_{i-1}$ dos not affect the optimal path p. Therefore, we can consider this fences as the whole plane and ignore them at all.

We can prove the lemma by induction on $i - j$. For $i = j$, the lemma follows from the definition of the i^{th}-component and i^{th}-term of the characteristic sequence. In both cases, it is the mark of the half-plane of $\bar{S}_i$ that contains x.

Now, suppose that the lemma holds for all $0 \leq i - j < l$. To prove the lemma for $i - j = l$, we first assume that x lies in α-side of $\bar{S}_i$. According to the definition, j^{th}-component of x is equal to the mark of the containing region of x in j^{th}-subdivision of the i^{th}-plane. As shows in Figure 8, let R be this region. The mark of this region in this subdivision is equal to the mark of the containing region of segment ab in j^{th}-subdivision of the $(i-1)^{th}$-plane. While x lies in α-side of $\bar{S}_i$ and our initial marking assignment has passing property, all extensions of S_j in this half-plane are exactly the extensions of S_j in the $(i-1)^{th}$-plane that intersect S_i

and by considering x as a point in the $i-1^{th}$-plane, it is still straightly reachable from S_j. This means that x and ab lie in the same region in j^{th}-subdivision of the $(i-1)^{th}$-plane and by induction the j^{th}-term of the characteristic sequence of x in the $(i-1)$-plane and its j^{th}-component are equal. On the other hand, the j^{th}-term of x in the i^{th}-plane is considered to be equal to the j^{th}-term of its characteristic sequence in the $(i-1)$-plane. These two equalities imply that j^{th}-term of the characteristic sequence of $x \in F_i$ is equal to its j^{th}-component.

When x lies in β-side of S_i, because of having passing property, all regions of the j^{th}-subdivision of the i^{th}-plane in this side are reflections of the regions in α-side of $\bar{S}_i$ in this subdivision, and the marks of corresponding regions are the same. This means that the j^{th}-component of x in β-side of $\bar{S}_i$ is equal to the j^{th}-component of the reflection of x on $\bar{S}_i$. Similarly, our algorithm computes the j^{th}-term of the characteristic sequence of the reflection of x on $\bar{S}_i$ as the j^{th}-term of x which follows the lemma in this case as well. □

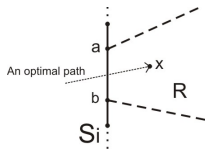

Figure 8: An optimal path to x from a point of ab.

Appendix 3: Algorithms

Algorithm 1 PChar(j,i,x)

```
if j = i then
    if x ∈ α-side of S̄_j then
        return α
    else
        return β
    end if
else
    if x lies in the α-side of S̄_i then
        return PChar(j,i − 1,x)
    else
        Let x′ be the reflection of x on S̄_i
        return PChar(j,i − 1,x′)
    end if
end if
```

Algorithm 2 MarkSides($\mathcal{S}$)

```
function MARK(S,i)
    if S_i completely lies on one side of S̄ then
        Mark the side of S̄ which contains S_i as β-side and the other as α-side.
    else
        if S lies in the α-side of S̄_i then
            MARK(S,i − 1)
        else
            Let S′ be the reflection of S on S̄_i
            MARK(S′,i − 1)
            Mark sides of S̄ the same as marks of their corresponding sides of S̄′.
        end if
    end if
end function
for i ← 1 to k do
    MARK(S_i,i − 1)
    Cut S_{i−1} by removing from it the part that lies on the α-side of S̄_i.
end for
```

Algorithm 3 OPT(i,x)

```
if i = 0 and x = s then
    return s.
end if
if x is an endpoint of S_i then
    return OPT(i − 1,x).
end if
Let C with sequence SQ(C) = (c_1, ..., c_l) be the cell with site c of SPM_i which contains x.
if the length of SQ(C) is zero then
    return <OPT(i,c),x >.
end if
Let j = l and L be a list with L =< x >.
while j ≠ 0 do
    Let I be the intersection of cx and S_{i−(l−j)}.
    Append I to the beginning of L.
    if x does not lie on the c_j-side of S_{i−(l−j)} then.
        Let c be the reflection of c on S̄_{i−(l−j)}.
    end if
    x = I.
    j = j − 1.
end while
return <OPT(i − l,c),L >.
```

Appendix 4: TLSP for intersecting line segments

In this section, we briefly describe how to extend our algorithm to solve TLSP when a segment $S_i \in \mathcal{S}$ may have intersection with S_{i+1}. Two types of intersections can be happened when S_i intersects S_{i+1}: *point intersection* and *interval intersection*. In point intersection, S_i and S_{i+1} have only one point in common but in interval intersection $S_i \cap S_{i+1}$

is a segment. If S_i and S_{i+1} overlap in interval I, we can simply remove S_i and S_{i+1} from $\mathcal{S}$ and replace them by segment I. So, we consider that there is no interval intersection between consecutive segments of $\mathcal{S}$.

Let $S_j = ab$ intersects S_{j+1} at point r. We define the extensions of S_j in the i^{th}-plane ($i \geq j$) as the union of extensions of ar and br in the i^{th}-plane (their interior is disjoint from S_{j+1}). So, If we mark the sides of S_j by α and β, each segment ar and br induces its own marking on the j^{th}-subdivision of the i^{th}-plane. Therefore, the mark of each region R of the j^{th}-subdivision of the i^{th}-plane is in $\{\alpha\alpha, \alpha\beta, \beta\alpha, \beta\beta\}$ which is composed of two letters. Assume that we have marked R as x_1x_2 where $x_1, x_2 \in \{\alpha, \beta\}$. If we replace S_j by ar, R completely lies in the x_1-region and if we replace S_j by rb, R completely lies in the x_2-region of the j^{th}-subdivision of the i^{th}-plane.

If we have characteristic sequences of all vertices and endpoints of segments, we can filter the wavelets in the second phase as follows: Let ω be a wavelet in F_{j-1} and $SQ(\omega)$ coincides $(x_{j-|SQ(\omega)|}, \ldots, x_{j-1})$. Assume that the j^{th}-term of the characteristic sequence of a point in $\mathcal{P}_j$ is $\alpha\beta$ (other cases are similar). This means that if ω intersects ar, it must generate a wavelet by passing or reflecting on ar in the α-side of S_j in F_j and if it intersects rb it must generate a wavelet by passing or reflecting on br in the β side of S_j in F_j. Note that we must also consider r as an initial site in F_j and its weight is computed by the wavefront of F_{j-1}. In both of the above cases the sequences of the generated wavelets in F_j are $(SQ(\omega), \alpha\beta)$.

Similar to Section 3, we use characteristic sequences and according to Lemma 2 if a point x is straightly reachable from S_{j-1}, the $(j-1)^{th}$-term of its characteristic sequence is equal to its $(j-1)^{th}$-component if we have a marking with passing property. When $S_j = ab$ intersects $S_{j+1} = cd$, two extensions in the j^{th}-plane may be intersected by S_{j+1} while their starting points lie on different sides of S_{j+1}. Figure 9 shows an example of such situations.

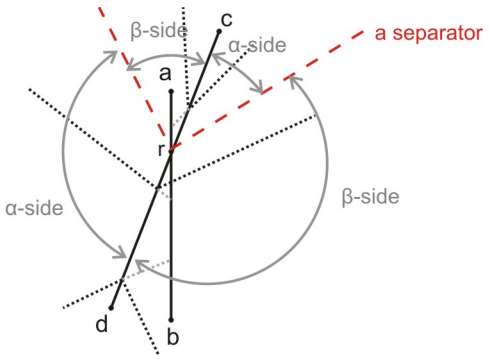

Figure 9: Extensions of S_j which intersect S_{j+1} are shown by gray dots and the extensions of S_{j+1} are shown by solid dots.

To handle such situations we extend the definition of α-side and β-side of S_{j+1} such that using this new definition, we can use procedure MarksSides and PChar to obtain characteristic sequences of vertices of fences and endpoints of segments.

Assume that function MARK marks the sides of cr and rd in such a way that the starting point of each extension in the j^{th}-plane which intersects these segments lies in β side of cr and rd. Lets consider a half-line from r which does not intersect any extension of S_l in the $(j+1)^{th}$-plane $(1 \leq l \leq j)$. We call this half-line and its reflection on $\bar{S}_{j+1}$ as *separators* (there is infinite separators of S_{j+1} but we need one in our algorithm) in the $(j+1)^{th}$-plane (separators are shown by dashed lines in Figure 9). These separators divide the plane into two regions namely R_1 and R_2. Lets R_1 be the one containing rc. We say that a point x or a segment S lies in α-side of S_{j+1} if it completely lies in α-side of rc and R_1 or α-side of rd and R_2. Otherwise, we say that it lies in β-side of S_{j+1}. By this definition, we see that all extensions in α-side of S_{j+1} coincide their corresponding extensions in the j^{th}-plane which is exactly the definition of the passing property. But, we need an extra clause in function MARK to handle the situation when a segment S intersects the separators in the $(j+1)^{th}$-plane: If a segment intersects a separator with starting point r in the $(j+1)^{th}$-plane, we mark the side of it which contains r as β-side and the other as α-side.

Therefore, in procedure MarkSides, when S_{j+1} intersects S_j we need to use function MARK for both cr and dr. Then, using a separator we can define its α and β-side. Now, because the marking obtained by MarkSides has the passing property, we can use procedure PChar to obtain the characteristic sequences of vertices of fences and endpoints of the segments.

Note that a separator for S_{j+1} from a point r in it can be easily obtained by considering r or its reflection recursively on α sides of $S_j, \ldots, S_1$ (Similar to procedure PChar) and in each step $l \leq j$ we maintain the bounds on the slope of a separator in which a separator does not intersect an extension of S_l.

According to the above discussion, if S_{j+1} intersects S_j, we need a linear time to obtain a separator for it. Also MarkSides performs function MARK at most $2k$ times. Also, the extra clause in MARK does not affect its complexity. Therefore, the running time of our algorithm remains $O(n^3k)$ for intersecting segments.