Visiting a Polygon on the Optimal Way to a Query Point[⋆]

Ramtin Khosravi[1] and Mohammad Ghodsi[2,3]

[1] School of Electrical and Computer Engineering, University of Tehran, P.O. Box 14395-515, Tehran, Iran
[2] Department of Computer Engineering, Sharif University of Technology, P.O. Box: 11365-9517, Tehran, Iran
[3] School of Computer Science, Institute for Studies in Theoretical Physics and Mathematics, P.O. Box: 19395-5746, Tehran, Iran
rkhosravi@ece.ut.ac.ir, ghodsi@sharif.ir

Abstract. We study a constrained version of the shortest path problem in polygonal domains, in which the path must visit a given target polygon. We provide an efficient algorithm for this problem based on the wavefront propagation method and also present a method to construct a subdivision of the domain to efficiently answer queries to retrieve the constrained shortest paths from a single-source to the query point.

1 Introduction

In this paper, we study the problem of finding a shortest path between two points inside a polygonal domain P (a simple polygon with a number of polygonal holes inside it) while the path is constrained to visit (i.e. has non-empty intersection with) a given *target polygon* T inside the free space. We call such a path a *shortest* T*-visiting path*. Our goal is to construct the *shortest* T*-visiting path map* $\mathrm{SPM}_T(s, P)$, a decomposition of the free space into a number of regions such that the combinatorial structure of the shortest T-visiting path from the given source point s to any point in a region is the same. This way, we will be able to find the length of the shortest T-visiting path between s and any query point in logarithmic time and report the actual path with an additional cost proportional to the complexity of the path. If the number of vertices in P and T be n and m respectively, and $N = m + n$, we preprocess the input (s, P, T) in $O(N \log N)$ time to construct a subdivision of the same space, so that the queries can be answered in $O(\log N)$ time.

Our method is based on the *continuous Dijkstra* paradigm to compute shortest paths in polygonal domains [4,10]. The main idea of our algorithm is to propagate a wavefront from s to visit T. Parts of T reached this way work as pseudo-sources for finding shortest T-visiting paths to points of the free space. We let those wavelets that first visit T propagate further and also propagate back a

[⋆] This work has been supported by a grant from IPM School of CS (No. CS1382-2-02).

© Springer-Verlag Berlin Heidelberg 2008

reflected version of those wavelets to cover points of the free space that shortest T-visiting paths to them visits the boundary of T and reflect back. This idea has been previously introduced in [7] by the authors.

A similar problem has been studied by the authors for the case of simple polygons [8] resulting in a linear algorithm for convex target polygons, as well as a method to construct the shortest T-visiting path map. The method presented here can be used to solve that problem for non-convex polygons. Extending the problem to multiple target polygons makes the problem similar to TSP with neighborhoods [2,3] which is NP-hard. Dror et al. [1] have presented an algorithm for the problem of finding the shortest path that visits k given convex polygons in a pre-specified order. Also, they have shown that the problem is NP-hard for the case of non-convex polygons.

We first introduce the concept of *reflective subdivision* in Sect. 2 which determines the structure of the shortest T-visiting paths without any obstacles in the plane. Then we extend this concept to the general case of polygonal domains in Sect. 3.

2 The Reflective Subdivision

To study the properties of the constrained shortest paths, we start by a simple case in which there is no obstacles in the plane. We are given a (possibly non-convex) *target polygon* T with m vertices and a point s outside T. We define $G(s)$ as the set of points where shortest T-visiting paths from s have their first intersections with T and call it the *gate* of s to T, or the gate of s for short. If T is a convex polygon, $G(s)$ is a connected chain on the boundary of T [1]. In general, $G(s)$ is not connected, but since we assumed there is no obstacles in the plane, there is at most one segment in $G(s)$ on any edge of T (which of course may be the entire edge).

Computing $G(s)$ can be easily done using algorithms for computing visibility polygon of a point in simple polygons [9,5], since if we consider T as an obstacle, $G(s)$ is the part of the boundary of T visible from s.

For an arbitrary point x, consider a shortest T-visiting path from s to x. If the segment $\overline{sx}$ intersects T, this segment is the desired path. The set of such points x makes a connected region called the *pass-through region*. If $\overline{sx}$ does not intersect T, the path consists of two segments $\overline{sc}$ and $\overline{cx}$ where c is a point on the boundary of T. We call c the *contact point*.

If c is an interior point of an edge e of T, the angle between $\overline{sc}$ and e is the same as the angle between e and $\overline{cx}$. We call the part of $G(s)$ on the interior of e an *edge-reflector*. If c is a vertex v of T, we call v a *vertex-reflector*. It easy to see that a vertex in $G(s)$ is a vertex-reflector if the angle made by the incident edges inside T is less than π. We define the *root* of a reflector r as r itself if it is a vertex-reflector, and s reflected about r if it is an edge-reflector.

Since the pass-through region can be easily computed and the corresponding shortest T-visiting paths are straight segments, we limit our attention to its complement, called D.

The *reflective subdivision* RS(s,T), or RS(s) for short, is the decomposition of D into faces such that the contact point of every point in a face is the same vertex-reflector r, or belongs to the same edge-reflector r. We call such a face a *reflective region* of r (Fig. 1).

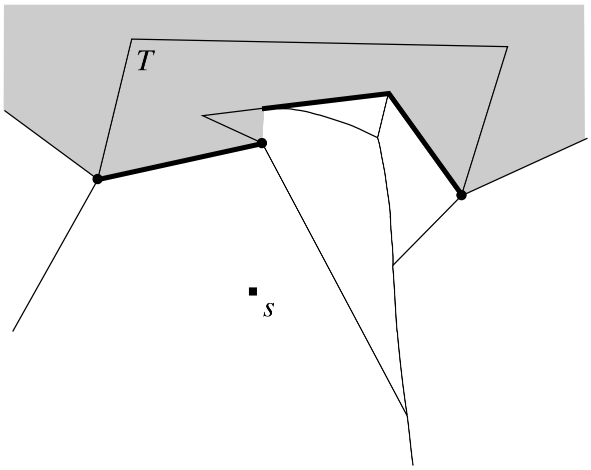

Fig. 1. The reflective subdivision RS(s,T): The shaded area is the pass-through region. The edge-reflectors are shown in thick segments and the vertex-reflectors are shown in black circles.

Some edges of RS(s) are from the boundary of D. Other edges separate reflective regions of different reflectors. In general, the edge separating two regions of reflectors r_1 and r_2 with roots a and b respectively, is defined by the bisector curve of a and b. This curve is a hyperbolic curve in general and is the locus of points x such that $w(a) + |\overline{ax}| = w(b) + |\overline{bx}|$ where $w(a)$ is $|\overline{sa}|$ if a is a vertex-reflector and zero if it is an edge-reflector. The following lemma establishes a linear bound on the size of this decomposition.

Lemma 1. *For a target polygon T with m vertices, the complexity of* RS(s,T) *is $O(m)$.*

Proof. We prove that for a reflector r, there is at most one reflective region. First observe that the points on a reflector r belong to its own reflective regions. Now consider a point x in a reflective region of r and assume c is the contact point of x. We can easily check that every point y on $\overline{cx}$ belongs to the same reflective region.

Now consider a case in which there is a reflector r with root a that has two reflective regions f_1 and f_2. Since these two regions are distinct, there exists a ray R emanated from a in the space between f_1 and f_2 such that every point on R from the intersection of R and r away from a belongs to the reflective regions of reflectors other than r. Let x be the intersection of R and r. Then, the length of the T-visiting path between s and x through r is the same as the length of such a path through another reflector namely r'. So, x lies on the bisector curve of the roots of r and r'. If this curve is a straight line, part of either f_1 or f_2 will be in the half-plane geodetically closer to r' which is impossible. The case that the curve is a not a straight line and has two intersections with r is not acceptable since part of r will reside in the reflective region of r'. So, there is at most one face in $\mathrm{RS}(s)$ corresponding to a reflector r, hence the number of faces is $O(m)$. The vertices of this subdivision are of these kinds: vertices of T, endpoints of edge-reflectors, and intersections between bisectors. The number of the first two kinds is $O(m)$. A vertex of the third kind borders at least three faces, hence the total number of vertices is $O(m)$. $\square$

We can compute $\mathrm{RS}(s)$ in $O(m \log m)$ time and $O(m)$ space using a simple sweep process. For a point $x \in D$, define $\delta(s, x)$ to be the length of the shortest T-visiting path from s to x. We sweep D based on the increasing value of δ. The sweep structure is a wavefront consisting of circular arcs (wavelets) centered at the roots of the reflectors. Initially, there will be a wavelet corresponding to each reflector. The *release time* for a vertex-reflector with root a is the length of $\overline{sa}$. At any instant during sweep, we say two bisectors are adjacent if they bound the same wavelet.

The only event in the sweep process occurs when the two bisectors separating the region of r from its two adjacent regions intersect. At this time the wavelet sweeping the region of r disappears and its two neighbors become adjacent. Since the intersections only occur between two adjacent bisectors, when processing an event, we can compute the times at which the newly created bisector intersects its two adjacent bisectors. It is easy to see that processing all $O(m)$ events can be done in $O(m \log m)$ time and $O(m)$ space.

3 Polygonal Domains

Let P be a polygonal domain having n vertices. A T-visiting path is a path in the free space having non-empty intersection with the target polygon T which we assume to have m vertices. We define the gate of s as before as the set of points where the shortest T-visiting paths from s have their first intersections with T. Again, $G(s)$ consists of a number of segments on the boundary of T.

Consider a maximally connected set of points on an edge e of T such that the last vertex on the shortest paths from s to them is the same vertex v of P. We call such a segment an edge-reflector and define its root as v reflected about e. Like before, a vertex of T in $G(s)$ is called a vertex-reflector and its root is the vertex itself. The edge-reflectors are a subset of the segments made on the boundary of T when intersected by $\mathrm{SPM}(s, P)$. In general, there can be $O(mn)$ such segments, but the following lemma shows that only $O(m + n)$ of these segments are edge-reflectors.

Lemma 2. *For a polygonal domain and a target polygon having n and m vertices respectively, there are $O(m + n)$ edge-reflectors.*

Proof. Let f be a cell of SPM(s) with root r. By adding three kinds of segments, we can decompose f into triangle-like regions (Fig. 2):

1. Segments connecting r to the intersection points between the boundaries of T and f,
2. segments connecting r to the vertices of f, and
3. segments each connecting r to some point on the boundary of f passing through a vertex of T inside f.

Since f is star-shaped with kernel r, all these segments are inside f and connect r to some point on the boundary of f. The regions obtained this way are either triangles, or bounded by two segments incident to r and a hyperbolic curve. Each region intersects a number of edges of T (possibly zero), but there is no vertices of T *inside* a region. Thus, the intersection of a region with T makes a number of segments with their end-points lying on the two boundary segments incident to r. Since the segments does not intersect inside the region, they can be ordered according to the increasing distance from r. It easy to check that only the nearest segment to r is a part of an edge-reflector. Since the shortest T-visiting paths to points on other segments already intersect it. Since there are at most $O(m + n)$ triangle-like regions in total, the number of edge-reflectors is bounded by the same order. □

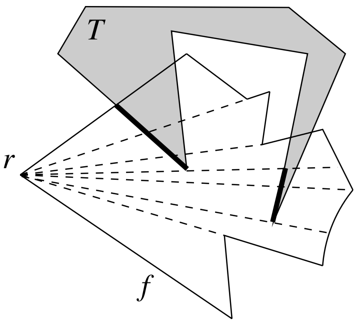

Fig. 2. Proof of lemma 2: The thick segments are parts of $G(s)$

To compute the set of reflectors, we can use the algorithms for constructing the shortest path map of a polygonal domain such as the algorithm of Hershberger and Suri [4] or that of Mitchell [10]. To do this, we consider T as an obstacle and define the polygonal domain $P' = P - T$. If we construct SPM(s, P'), the boundary of T is partitioned into a number of segments. Some of these segments are edge-reflectors. Consider a segment that is made by the SPM cell with root r. If r is a vertex of T, then the segment cannot be a part of $G(s)$. Assuming r is a vertex of P, we locate r in the two shortest path trees SPT(s, P) and SPT(s, P'). If the path from s to r is the same in both trees, considering T as an obstacle has no effect in the shortest path to the points in the segment under consideration, so it is an edge-reflector. So, computing the edge-reflectors can be done in $O((m+n)\log(m+n))$ time and the same space. This computation also produces a list of vertex reflectors.

We define the pass-through region as before. Let D be the free space with the pass-through region removed. $\mathrm{RS}(s,T)$ is the partition of D into regions according to the reflector that is first visited along shortest T-visiting paths from s. A similar argument as the one in lemma 1 proves there are $O(m)$ regions in the subdivision and its complexity is $O(m+n)$.

Computing the shortest T-visiting path map $\mathrm{SPM}_T(s,P)$ can be done using wavefront propagation method. This map has two parts: one corresponding to the pass-through region, and another for D. The first part is $\mathrm{SPM}(s,P)$ restricted to the pass-through region. For the second part, we have multiple sources which are the roots of the reflectors. Each source has a specified release-time. For vertex-reflectors, the release time is the geodesic distance from s to that vertex, and for the edge-reflectors, it is the geodesic distance from s to the last vertex v on the shortest paths from s to points on the edge-reflector, plus d which is the distance from v to the reflector segment. To cover points in D, we use a wavefront propagation algorithm to "reflect back" those parts of the original wavefront started from s that have visited T. Note that the initial wavelets are to be computed carefully, since some sources may lie outside D. For an edge-reflector, if v is the last vertex on the shortest path from s to points on the reflector, the initial wavelet is centered at the $\bar{v}$ which is v reflected about the edge-reflector, and the radius is d.

Both algorithms of [4] and [10] are capable of handling multiple source with specified release-times. If we use the first algorithm (that of Hershberger and Suri) which is worst-case optimal, we obtain $O((m+n)\log(m+n))$ time and space bounds to construct $\mathrm{SPM}_T(s,P)$. Since computing the reflectors can be done using the same algorithm, the order remains the same for the entire computation. Hence we have our main result as follows.

Theorem 1. *For a polygonal domain P and a target polygon T inside P with N vertices in total, and a source point s, we can compute the shortest T-visiting path map $\mathrm{SPM}_T(s,P)$ in $O(N\log N)$ time and space.*

4 Conclusion

We showed how one can use the wavefront propagation method to partition the free space in a polygonal domain according to the combinatorial structure of shortest paths from a given source point s to the points in the free space that has non-empty intersection with a target polygon T. We showed how to compute this subdivision having an algorithm for wavefront propagation capable of handling multiple sources with specified release-times. The best known method so far ([4]) solves this problem in $O((m+n)\log(m+n))$ time and space.

We leave an open problem that is whether one can use the methods based on searching the visibility graph to find the shortest T-visiting path between two points. This is particularly important, since the best known algorithm using this method by Kapoor et al. [6], solves the shortest path problem in polygonal domains in $O(n + h^2 \log n)$ which is only linear in n, while being quadratic in the number, h, of holes.

References

1. Dror, M., Efrat, A., Lubiw, A., Mitchell, J.S.B.: Touring a sequence of polygons. In: Proc. 35th ACM Sympos. Theory Comput (2003)
2. Dumitrescu, A., Mitchell, J.S.B.: Approximation algorithms for TSP with neighborhoods in the plane. In: Symposium on Discrete Algorithms, pp. 38–46 (2001)
3. Gudmundsson, J., Levcopoulos, C.: A Fast Approximation Algorithm for TSP with Neighborhoods and Red-Blue Separation. In: Asano, T., Imai, H., Lee, D.T., Nakano, S.-i., Tokuyama, T. (eds.) COCOON 1999. LNCS, vol. 1627, pp. 473–482. Springer, Heidelberg (1999)
4. Hershberger, J., Suri, S.: An optimal algorithm for Euclidean shortest paths in the plane. SIAM J. Comput. 28(6), 2215–2256 (1999)
5. Joe, B., Simpson, R.B.: Correction to Lee's visibility polygon algorithm. BIT 27, 458–473 (1987)
6. Kapoor, S., Maheshwari, S.N., Mitchell, J.S.: An efficient algorithm for euclidean shortest paths among polygonal obstacles in the plane. Discrete Comput. Geom. 18, 377–383 (1997)
7. Khosravi, R., Ghodsi, M.: Shortest paths in polygonal domains with polygon-meet constraints. In: Proc. 19th European Workshop Comput. Geom., pp. 137–142 (2003)
8. Khosravi, R., Ghodsi, M.: Shortest paths in simple polygons with polygon-meet constraints. Inform. Process. Lett. 91, 171–176 (2004)
9. Lee, D.T.: Visibility of a simple polygon. Comput. Vision Graph. Image Process 22, 207–221 (1983)
10. Mitchell, J.S.B.: Shortest paths among obstacles in the plane. Internat. J. Comput. Geom. Appl. 6, 309–332 (1996)