# Shortest Paths in Polygonal Domains with Polygon-Meet Constraints

(Extended Abstract)

Ramtin Khosravi* Mohammad Ghodsi†

Department of Computer Engineering
Sharif University of Technology
P.O. Box: 11365-9415,
Tehran, Iran.
Tel: +98 (21) 600-5616

**Abstract**

In this paper, we study the problem of finding the shortest path in a polygonal domain in which the path should meet (touch or cross) a simple polygon in the domain. Our method uses the continuous Dijkstra paradigm and the idea of wavelet reflection and solves the problem in worst-case optimal time $O(n \log n)$.

## 1 Introduction

We study a special kind of constraints in the shortest path problem called polygon-meet constraints, in which the shortest path from source to destination should meet a given polygon in a polygonal domain. By meeting a polygon, we mean the path should have non-empty intersection with the closure (interior plus boundary) of a polygon, i.e., either touch the boundary of the polygon and reflect, or cross through a part of the polygon.

Our approach is to use the *continuous Dijkstra* paradigm for finding the shortest polygon- meeting path between two points. As a result of using this paradigm, a subdivision of the domain can be built to answer single-source shortest polygon-meeting path queries in $O(\log n)$ time. The idea of the method is to propagate a wavefront from the source until it touches the boundary of the target polygon. Upon touching the boundary, the original wavefront is marked as *met* and is propagated further while ignoring future contacts with the target polygon. Besides the original wavefront, its *reflected* version is also propagated in the opposite direction which is also marked as *met*. By *met* wavefront we mean the set of points in the free space to which the length of the shortest polygon-meeting path is $\delta$. The reflected part corresponds to those point whose shortest polygon-meeting path from source has been touched the boundary of the target polygon and has been reflected. If we use the method of Hershberger and Suri [1] for wavefront propagation, we obtain a time-bound of $O(n \log n)$ and $O(n \log n)$ space for our problem which is worst-case optimal. As the size of the constructed map is linear, each query can be answered in $O(\log n)$ time. To the best of our knowledge, no other result has been available on this problem.

## 2 Computing Polygon-Meeting Paths

Let $s$ be the source point and $\mathcal{P}$ be the target polygon to be met by the path. The total number of vertices in the obstacle polygons as well as the target polygon is assumed to be $n$. Given a query point $t$, our goal is to find the shortest path between $s$ and $t$ which residing completely in the free space, having non-empty intersection with the closure of the target polygon $\mathcal{P}$.

*ramtin@mehr.sharif.edu
†ghodsi@sharif.edu

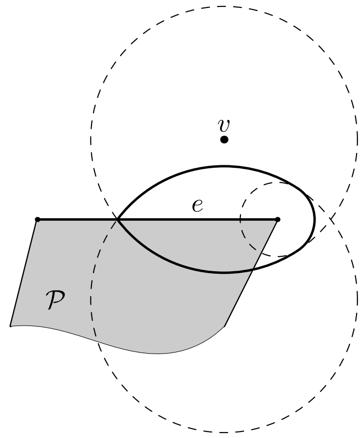


Figure 1: Three kinds of *met* wavelets are shown using solid arcs. The lower arc is a part of the original wavefront generated by $v$, the upper arc is its reflected version, and the right arc (part of the small circle) is generated upon collision of the original wavefront by the right end-point of the edge $e$ of $\mathcal{P}$.

Suppose we have a line $l$, and a source point $x$ in the plane (with no obstacles). Let $H$ be the half-plane generated by $l$ in which $x$ lies. For a point $y \in H$, we define the shortest reflective distance from $x$ to $y$, as the length of the shortest path that starts from $x$, meets some points of $l$, and continues to the point $y$. It is easy to see that the locus of points $y$ that are of shortest reflective distance $\delta$ from $x$ is the part of a circle of radius $\delta$ and center $\overline{x}$ that lies in $H$, where $\overline{x}$ is the reflection of $x$ about $l$. If we consider a segment $e$ instead of the line $l$, but with the same requirement for the shortest path, there will be two additional circular arcs generated from the two end-points of $e$.

Based on the above observations, we use the idea of reflecting a wavelet when colliding the boundary of $\mathcal{P}$, as well as letting it continue inside $\mathcal{P}$. By *original* wavefront we mean the wavefront resulting from the shortest path algorithm, ignoring the target polygon $\mathcal{P}$. Original wavelets are defined similarly. Upon collision of an original wavelet with the boundary of $\mathcal{P}$, a met wavelet of one of these kinds may be generated (Figure 1):

1. A wavelet that is essentially the part of the original wavelet that enters $\mathcal{P}$.
2. A wavelet that is the reflection of the original wavelet about the edge of $\mathcal{P}$ with which the collision is occured.
3. A wavelet of zero radius. This happens when the original wavelet collides a vertex of $\mathcal{P}$.

We call these kind of wavelets (which are generated right after collision of original wavelets with the boundary of $\mathcal{P}$), *primary met wavelets*. In all of the above three forms, the primary wavelet is a circular arc: for the first kind, the center of the arc is the same as that of original wavelet, for the second kind, it is the center of the original wavelet reflected about the boundary edge, and for the third kind, it is the boundary vertex. Right after generation, primary met wavelets continue to propagate according to known rules of wavefront propagation in the Continuous Dijkstra paradigm. New wavelets may be added to the wavefront upon collision of met wavefronts with obstacle vertices, or may be deleted upon collision of met wavefront with each other. We call the newly generated wavelets, *secondary met wavelets*. Note that the collision of met wavelets with original wavelets are not considered during simulation.

**Lemma 2.1** *The number of primary met wavelets is $O(n)$.*

By *shortest $\mathcal{P}$–meeting path map* ($\mathcal{P}$-SPM for short), we mean the decomposition of the free space of the polygonal domain $\mathcal{P}$ to cells such that the shortest $\mathcal{P}$-meeting path to all points in one cell has the same combinatorial structure. Corresponding to two types of met wavelets (primary and secondary), there are two kinds of cells in the $\mathcal{P}$-SPM, namely primary and secondary cells. Boundary of all primary cells have non-empty intersection with the boundary of $\mathcal{P}$. This intersection may be a segment, or a single vertex of $\mathcal{P}$.

**Lemma 2.2** *The shortest polygon-meeting path map ($\mathcal{P}$-SPM) has linear size.*

Finally, note that unlike the wavefront in the original shortest path algorithm, the met wavefront in our algorithm may be disconnected, i.e. consists of multiple closed chains of circular wavelets.

To compute $\mathcal{P}$-SPM, we use two extensions of the algorithm of [1] that the authors have mentioned: non-point sources and multiple sources with specified release times. The algorithm runs in two phases:

1. Computing the primary met wavelets.

2. Propagating the met wavefront.

In order to compute the primary met wavelets, we first use the original wavefront propagation algorithm to find the times at which the original wavefront meets $\mathcal{P}$ boundary generating primary met wavelets. To do this, we first consider $\mathcal{P}$ as an obstacle in the domain. This prevents original wavelets to enter the interior of $\mathcal{P}$. At the end of this phase, the following information should be obtained and stored:

1. The first time at which each wavelet collides $\mathcal{P}$ boundary. Note that if more than one wavelet collide a boundary edge, the times of all collisions should be stored.

2. The first time at which a wavelet collides a $\mathcal{P}$ vertex.

Note that considering $\mathcal{P}$ as an obstacle may cause some of the generated time labels be greater than the actual time if we let the wavefront enter $\mathcal{P}$ interior, but this does not makes any problem, since in the second phase, when the simulation time reaches a time label, we check whether the generator of the event (either a $\mathcal{P}$ edge or vertex) is already met. In this case we simpley ignore the event. In the second phase, we consider the problem as an extension of the standard shortest path algorithm in which there are multiple non-point sources with specified release time. Since the met wavefront may have several independent components, we should consider the propagation in the multiple-source model. As the times the primary met wavelets are generated may be different, these component have different release times. Considering the algorithm based on the non-point extension is due to the fact that the first two kind of primary wavelets which are generated as a result of collision of original wavelets and $\mathcal{P}$ edges has non-zero radius when generated. This requires special attention for initialization of these wavelets.

Using the above schema for the algorithm, we can conclude our main result:

**Theorem 2.1** *For a polygonal domain of total complexity $n$, and a target polygon of size $O(n)$, the shortest polygon-meeting path map of the domain can be built in time $O(n \log n)$ and $O(n \log n)$ space.*

**Proof.** Since the target polygon has $O(n)$ edges, the first phase of our algorithm can be done in $O(n \log n)$ time and space using standard shortest path algorithm. The event queue can be also initialized in $O(n \log n)$ time. Since the total number of met wavelets is bounded by $O(n)$ and the size of the resulting map is also linear, the time required to construct the map is $O(n \log n)$ using extensions of the standard algorithm in [1]. $\square$

# 3 Conclusions

We presented an algorithm for finding a special kind of shortest path map, in which the path is constrained to meet a target polygon, which may be a visibility polygon or a resource area in practical applications. The algorithm was based on a variation of the Continuous Dijkstra method for constructing shortest path maps. It is possible to study the problem in several extensions. One exension is to consider different metrics for distance calculations, such as link–distance.

Another extension is to consider the problem in other domains such as simple polygons or polyhedral surfaces. The case of simple polygon is particularly interesting since existence of linear time algorithms for finding shortest paths arises the question of solving the problem in linear time. Extending the method discussed in this paper to the case of polyhedral surfaces is also interesting since the method of Kapoor [2] for finding the shortest path in that domain is based on the Continuous Dijkstra paradigm.

If we consider the problem in the polygonal domain with the extension that the path should meet several polygons, it is called *TSP with neighborhoods* problem which is NP-hard and is studied in [3], but if we fix the order of meeting the target polygons, the problem seems to be less complex, yet we do not have any result for it.

# References


[1] John Hershberger and Subhash Suri. An optimal algorithm for Euclidean shortest paths in the plane. Manuscript, Washington University, 1995.

[2] S. Kapoor. Efficient computation of geodesic shortest paths. In *Proc. 32th Annu. ACM Sympos. Theory Comput.*, pages 770–779, 1999.

[3] C. Mata and J. S. Mitchell. Approximation algorithms for geometric tour and network design problems. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 360–369, 1995.