Discrete Comput Geom 18:377–383 (1997)

Discrete & Computational
Geometry

An Efficient Algorithm for Euclidean Shortest Paths Among Polygonal Obstacles in the Plane

S. Kapoor,[1] S. N. Maheshwari,[1] and J. S. B. Mitchell[2]

[1]Department of Computer Science and Engineering, Indian Institute of Technology,
Hauz Khas, New Delhi, India
{skapoor, snm}@cse.iitd.ernet.in

[2]Department of Applied Mathematics and Statistics, State University of New York,
Stony Brook, NY 11794–3600, USA
jsbm@ams.sunysb.edu

Abstract. We give an algorithm to compute a (Euclidean) shortest path in a polygon with h holes and a total of n vertices. The algorithm uses $O(n)$ space and requires $O(n+h^2 \log n)$ time.

1. Introduction

Let P denote a (multiply connected, closed) polygon in the plane having h holes ("obstacles") and a total of n vertices. The problem of computing a shortest ("geodesic") path from a point $s \in P$ to a point $t \in P$ has been well studied in computational geometry; see [1], [4], [6], [8], [10], [13], [14], [17], and [21]–[23], as well as the recent survey chapter by Mitchell [15]. In the case that $h = 0$, the shortest path can be computed in $O(n)$ time [6], [12]. In the case that $h > 0$, the complexity of the problem has been worst-case quadratic ($O(n^2)$), until the recent $O(n^{1.5+\varepsilon})$ algorithm of Mitchell [14], which develops the "continuous Dijkstra" paradigm, and its improvement by Hershberger and Suri [7], [8], which results in an algorithm whose running time is $O(n \log n)$.

A lower bound of $\Omega(n + h \log h)$ is easy to establish (from sorting or convex hulls), but, to date, there is no matching upper bound. In fact, no upper bounds of the form $O(n + f(h))$, having *linear* dependence on n, have previously been published.

Here, we offer a simple algorithm whose dependence on n is linear, both in time and space. The time dependence on h, however, is slightly worse than quadratic: $O(n + h^2 \log n)$. Thus, while our algorithm is optimal for values of h that are roughly $O(\sqrt{n})$, it is inferior to the best known methods in cases in which h is not relatively small compared with n. In many applications (e.g., geographic problems), though, h may be expected to be small compared with n.

Our approach is to construct a relevant subgraph of the visibility graph of P, augmented with path information obtained from the "corridor" structure of P. We search the graph using Dijkstra's algorithm, constructing edges as the algorithm proceeds, in order to keep the space complexity to $O(n)$.

A preliminary version of this paper appeared in part in [10]. Here, we give a somewhat simplified variation of the original methods of [10]. Full details of the original approach, which leads to some slightly improved time bounds (see the remarks after Theorem 1), can be found in [11].

2. The Algorithm

We assume that P is a closed, connected set, whose boundary consists of n line segments, having $h + 1$ connected components: the *outer boundary*, plus the boundaries of each of the h polygonal holes (*obstacles*) of P. Without loss of generality, P is bounded, and we assume that s and t are interior to P.

We triangulate P; this can be done in time $O(n + h \log^{1+\varepsilon} h)$ [2]. We then incorporate s and t into the triangulation by linking s (resp. t) to the three corners of the triangle, τ_s (resp. τ_t), that contains it. (We assume that $\tau_s \neq \tau_t$; otherwise, the shortest path from s to t is simply the line segment joining them.) Let $\mathcal{T}$ denote the resulting triangulation, and let $\mathcal{G}_\mathcal{T}$ denote the graph-theoretic dual of $\mathcal{T}$. Note that $\mathcal{G}_\mathcal{T}$ is a planar graph having $O(n)$ nodes, $O(n)$ arcs, and $h + 1$ faces, and that at least one of the three nodes dual to the triangles incident on each of s and t has degree 3.

Kapoor and Maheshwari [10] and Mitchell and Suri [16] have shown how contraction of $\mathcal{G}_\mathcal{T}$ naturally leads to a decomposition of P into "corridors," as follows: First, we take any degree-1 node of $\mathcal{G}_\mathcal{T}$ and delete it, along with its incident edge; we repeat until there are no degree-1 nodes. Now, $\mathcal{G}_\mathcal{T}$ has $h+1$ faces and all nodes are of degree 2 or 3. Assume now that $h \geq 2$; then not all nodes are of degree 2, implying that there are at least two degree-3 nodes (since there must always be an even number of odd-degree vertices in a graph). Next, for each degree-2 node, we delete it and replace its two incident edges with a single edge. The resulting graph, call it $\mathcal{G}$, is a 3-regular planar graph, possibly with loops and possibly with multiedges (two edges joining the same pair of nodes). Further, $\mathcal{G}$ has $h + 1$ faces, $2h - 2$ nodes, and $3h - 3$ arcs (by Euler's formula). Each node of $\mathcal{G}$ corresponds to a triangle in $\mathcal{T}$; we call these $O(h)$ triangles the *junction triangles*. Removal of the junction triangles from P results in a set of simple polygons, one for each arc of $\mathcal{G}$, which we refer to as the *corridors* of P. (That corridors are simply connected follows immediately from the contraction process.) Refer to Fig. 1.

The boundary of a corridor C consists of four pieces: (1) a polygonal chain along the boundary of an obstacle O_1, from a vertex a to a vertex b; (2) a diagonal (junction triangle) edge from b to a vertex c (possibly $O_2 = O_1$); (3) a polygonal chain along the boundary of O_2, from c to a vertex d; and (4) a diagonal (junction triangle) edge from d back to a. The segments ad and bc are the *doors* of C. (Note that a may equal b, or c may equal d, if C corresponds to a loop in $\mathcal{G}_\mathcal{T}$.) If we replace the paths from a to b and from c to d with their geodesic paths, π_{ab} and π_{cd}, within C (imagine pulling "taut"

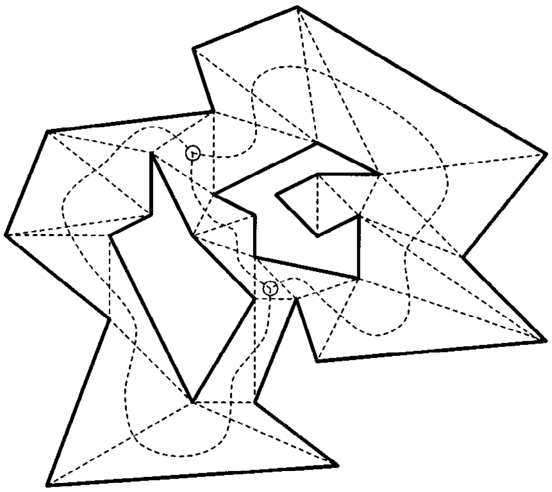

Fig. 1. A polygon P having two holes. There are two junction triangles (shown with small circles in them), linked by three arcs of $\mathcal{G}$ (shown with dashed curves). Removal of the two junction triangles results in three corridors.

the paths), then we obtain a region, called an *hourglass*, $H \subseteq C$, with the boundary consisting of π_{ab}, π_{cd}, and the doors ad and bc; see [5], [6], and [12]. H is called an *open hourglass* if $\pi_{ab} \cap \pi_{cd} = \emptyset$ (i.e., if there exists a line segment joining the doors of C, separating O_1 and O_2); H is called a *closed hourglass* otherwise. Refer to Fig. 2. If H is open, then the paths π_{ab} and π_{cd} are concave with respect to H; i.e., each vertex of these paths has an internal angle greater than 180°. If H is closed, then it consists of two "funnels," with apices x and y, and a geodesic path, $\pi_C = \pi_{ab} \cap \pi_{cd}$, joining x and y; we call π_C the *corridor path* for C. By the results of [6] and [12], we can compute H from C in linear time, by computing the paths π_{ab} and π_{cd}; doing this for all corridors requires only $O(n)$ time.

Let $Q \subseteq P$ denote the union of the junction triangles and the hourglasses of P. Thus, Q consists of junction triangles, open hourglasses, funnels, and corridor paths. Let Q' denote the set Q, minus the corridor paths. (Note that Q' may be disconnected.) There are only $O(h)$ junction triangles and $O(h)$ funnel apices. Thus, the boundary of Q' consists of $O(h)$ convex vertices and $O(h)$ reflex chains.

Let π^* be a shortest s-t path in P. If π^* intersects a corridor C, it must intersect both doors of C (if it enters and leaves through the same door, it could be shortened), and stay within the corresponding hourglass (again, by local optimality). Furthermore, $s, t \in Q$, since, by construction of $\mathcal{T}$, at least one of the three triangles incident on each of s and t has degree 3 in $\mathcal{G}_{\mathcal{T}}$, and so must be a junction triangle, which survives after contraction. We conclude:

Lemma 1. $\pi^* \subset Q$.

Let $VG(Q')$ denote the *visibility graph* of Q', whose node set consists of all vertices of Q' and whose edge set corresponds to pairs of vertices for which the connecting (open)

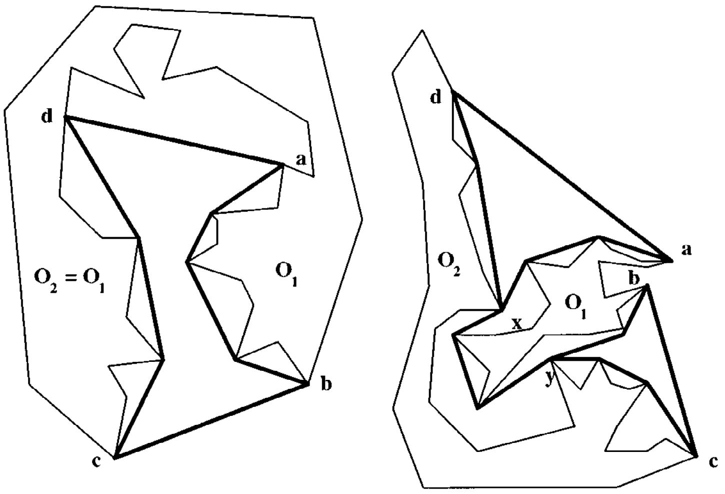

Fig. 2. An example of an open hourglass (left) and a closed hourglass (right), having a corridor path, π_C, linking the apices x and y.

line segment lies within the interior of Q'. We say that an edge of $VG(Q')$ is *relevant* if it is locally tangent to the boundary of Q' at each of its endpoints. (By definition, each edge on the boundary of Q' is not a relevant visibility graph edge.) By standard local optimality arguments, we see that $\pi^* \cap Q'$ must consist of relevant visibility graph edges, together with convex chains on the boundary of Q'. Thus, we are motivated to compute the relevant visibility graph edges for Q'.

Now, the boundary of Q' consists of $O(h)$ vertices that are convex with respect to Q', and $O(h)$ reflex (with respect to Q') chains. This means that the complementary region, $R = \Re^2 \backslash Q'$, consists of a set of polygons having a total of $O(h)$ *reflex* vertices, and $O(h)$ *convex* chains. Thus, R can be partitioned into $O(h)$ convex polygons (e.g., by extending an angle-bisecting segment inward from each reflex vertex, as done by Hertel and Mehlhorn [9]). These convex polygons have pairwise-disjoint interiors.

We devise a method of computing the relevant visibility graph edges based on the methods of Rohnert [19], [20], who studied the problem of computing shortest paths among convex polygonal obstacles. Indeed, other than the presence of corridor paths, we have reduced the general shortest-path problem to that of shortest paths among convex obstacles.

First, note that for any pair of convex polygonal obstacles, there are four tangent (supporting) segments, which can be computed in time $O(\log n)$. For any one convex polygon, there are $O(h)$ incident tangent segments. Those tangent segments that are "visible" (not penetrating any obstacle) can be identified by a simple sweep (by slope) in time $O(h \log h)$; the $O(h)$ tangent segment endpoints partition the boundary of the convex polygon into $O(h)$ pieces (convex chains), the lengths of which are easily tabulated within the same $O(h \log h)$ time bound. (These pieces can be considered as single "edges" for purposes of computing shortest paths.) If we were to compute all of $VG'(Q')$ at once, this would require $O(n + h^2 \log n)$ time and $O(h^2 + n)$ space; we can reduce the space requirement to $O(n)$ by computing the segments of $VG'(Q')$ *as they become needed in an execution of Dijkstra's shortest-path algorithm* (from source point s), as is done in [20].

The lengths of the corridor paths are all computed within total time $O(n)$. Each corridor path can be treated as a single "edge" during the execution of Dijkstra's algorithm. In all, there are $O(h^2)$ edges of $VG'(Q')$, $O(h^2)$ pieces of obstacle boundaries (induced by the endpoints of the $VG'(Q')$ edges), and $O(h)$ corridor edges. The resulting graph that is searched therefore has $O(h^2)$ edges and $O(h^2)$ nodes, and thus can be searched in time $O(h^2 \log h)$ using Fredman and Tarjan's implementation of Dijkstra's algorithm [3]. In addition to this time complexity is the $O(n + h^2 \log n)$ overhead in the construction of $VG'(Q')$ and the $O(n + h \log^{1+\varepsilon} h)$ time for triangulating P. In conclusion:

Theorem 1. *A Euclidean shortest path in a polygon having n vertices and h holes can be computed in time $O(n + h^2 \log n)$, using $O(n)$ space.*

Remarks. 1. The time spent finding tangents in the above algorithm can be reduced to $O(h^2 \log(n/h))$ by a simple observation, pointed out by a referee: the n vertices of the polygon are partitioned among the $O(h)$ convex chains, so the time required to compute a common tangent is not just $O(\log n)$, but is $O(\log n_i + \log n_j)$, where n_i and n_j are the chain sizes. The total time spent computing all tangents is then

$$O\left(\sum_{i \neq j} \log n_i + \log n_j\right) = O\left(h \sum_i \log n_i\right) \leq O(h^2 \log(n/h)).$$

Since we sort the tangents, the overall time for computing the relevant visibility graph is still $O(n + h^2 \log(h + n/h)) = \Theta(n + h^2 \log n)$. However, this sorting step can also be eliminated, at the expense of increased space complexity (equal to the number of edges of the visibility graph), by using the "visibility complex," as introduced by Pocchiola and Vegter [18]. This reduces the visibility graph construction time to $O(n + h^2 \log(n/h))$. In fact, it may be possible to use the visibility complex to reduce the construction time to $O(n + e \log n)$, where e is the size of the relevant visibility graph. We leave this question open. Note, however, this approach would sacrifice some of the simplicity of the method we have proposed here.

2. An alternative improvement has been obtained by Kapoor and Maheshwari [11], who obtain a time bound of $O(n + e \log e + h^2 \log(e/h) + h^2 \log(n/h))$, based on an algorithm that propagates visibility information through corridors and across junctions; refer to the brief outline below, and see the full paper [11] for details. Their time bound is sensitive to the number, e, of edges in the relevant visibility graph, $VG'(Q')$. Their algorithm uses only $O(n)$ space. Note that $e = O(h^2)$.

A Visibility Propagation Algorithm

Here, we briefly outline the visibility propagation algorithm detailed in [10] and [11] for computing the relevant visibility graph.

The input to the algorithm is the set of corridors, which have been identified as previously described. The goal now is to identify the (visible) common tangents between pairs of convex chains that bound the hourglasses corresponding to corridors. For each convex chain C, visible tangents to other convex chains are constructed by a propagation algorithm that maintains sectors of visible regions—"visibility ranges," each determined by two tangents between the chain C and other convex chains. The visibility range represents the region from which a visible tangent may be drawn to the chain C. As we sweep outward from the initial convex chain C, we traverse corridors and junctions and the visibility ranges change accordingly. To help in processing, the ranges for a chain C are maintained in sorted order in a balanced tree which allows for deletion, insertion, and merges of ranges along with updates and searches. (Since the chain C is convex the tangents incident onto it can be sorted in order of the vertices of the chain to which they are incident.) At each step of the algorithm, the visibility ranges present at a door of a corridor are propagated according to two cases: (1) propagation through a corridor, which breaks into subcases according to whether the corresponding hourglass is open or closed; and (2) propagation across a junction triangle, which results in either a split of a set of ranges or a merging of two sets of visibility ranges. Since there are only $O(h)$ corridors and junction triangles, and each operation on a set of ranges can be accomplished in $O(\log n)$ time, the resulting algorithm yields the relevant visibility graph in time $O(h^2 \log n)$. As mentioned above, with some further care, it is possible to modify the propagation algorithm to be sensitive to the size, e, of the resulting visibility graph; see [11].

3. Conclusion

We leave as a challenging open problem the question of whether or not shortest paths can be computed in time $O(n + h \log h)$, using $O(n)$ space.

Acknowledgments

We thank the referees for several useful suggestions that improved this paper.

References

1. T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai, Visibility of disjoint polygons, *Algorithmica* **1** (1986), 49–63.
2. R. Bar-Yehuda and B. Chazelle, Triangulating disjoint Jordan chains, *Internat. J. Comput. Geom. Appl.* **4** (1994), 475–481.
3. M. Fredman and R. Tarjan, Fibonacci heaps and their uses in improved network optimization problems, *J. Assoc. Comput. Mach.* **34** (1987), 596–615.
4. S.K. Ghosh and D.M. Mount, An output-sensitive algorithm for computing visibility graphs, *SIAM J. Comput.* **20** (1991), 888–910.
5. L.J. Guibas and J. Hershberger, Optimal shortest path queries in a simple polygon, *J. Comput. System Sci.* **39** (1989), 126–152.

6. L.J. Guibas, J. Hershberger, D. Leven, M. Sharir, and R. Tarjan, Linear time algorithms for visibility and shortest path problems inside triangulated simple polygons, *Algorithmica* **2** (1987), 209–233.
7. J. Hershberger and S. Suri, On computing Euclidean shortest paths in the plane, *Proc*. 34*th Annual IEEE Symposium on Foundations of Computer Science*, 1993, pp. 508–517.
8. J. Hershberger and S. Suri, An optimal-time algorithm for Euclidean shortest paths in the plane, Manuscript, Washington University, 1995.
9. S. Hertel and K. Mehlhorn, Fast triangulation of the plane with respect to simple polygons, *Inform*. *Control* **64** (1985), 52–76.
10. S. Kapoor and S.N. Maheshwari, Efficient algorithms for Euclidean shortest path and visibility problems with polygonal obstacles, *Proc*. *Fourth Annual ACM Symposium on Computational Geometry*, 1988, pp. 172–182.
11. S. Kapoor and S.N. Maheshwari, An efficient algorithm for Euclidean shortest path with polygonal obstacles, Technical Report, Dept. of Computer Science and Engineering, Indian Institute of Technology, Hauz Khas, New Delhi, 1994.
12. D.T. Lee and F.P. Preparata, Euclidean shortest paths in the presence of rectilinear boundaries, *Networks* **14** (1984), 393–410.
13. J.S.B. Mitchell, A new algorithm for shortest paths among obstacles in the plane, *Ann*. *Math*. *Artificial Intel*. **3** (1991), 83–106.
14. J.S.B. Mitchell, Shortest paths among obstacles in the plane, *Internat*. *J. Comput*. *Geom*. *Appl*. **6** (1996), 309–332.
15. J.S.B. Mitchell, Shortest paths and networks, in *CRC Handbook of Discrete and Computational Geometry* (E. Goodman and J. O'Rourke, eds.), CRC Press, Boca Raton, FL, 1997, pp. 445–466.
16. J.S.B. Mitchell and S. Suri, Separation and approximation of polyhedral surfaces, *Comput*. *Geom*. *Theory Appl*. **5** (1995), 95–114.
17. M.H. Overmars and E. Welzl, New methods for computing visibility graphs, *Proc*. *Fourth Annual ACM Symposium on Computational Geometry*, 1988, pp. 164–171.
18. M. Pocchiola and G. Vegter, The visibility complex, *Internat*. *J. Comput*. *Geom*. *Appl*. **6** (1996), 279–308.
19. H. Rohnert, Shortest paths in the plane with convex polygonal obstacles, *Inform*. *Process*. *Lett*. **23** (1986), 71–76.
20. H. Rohnert, A new algorithm for shortest paths avoiding convex polygonal obstacles, Report A86/02, Fachber. Inform., Univ. Saarlandes, Saarbrücken, 1986.
21. M. Sharir and A. Schorr, On shortest paths in polyhedral spaces, *SIAM J. Comput*. **1** (1986), 193–215.
22. J.A. Storer and J.H. Reif, Shortest paths in the plane with polygonal obstacles, *J. Assoc*. *Comput*. *Mach*. **41** (1994), 982–1012.
23. E. Welzl, Constructing the visibility graph for n line segments in $O(n^2)$ time, *Inform*. *Process*. *Lett*. **20** (1985), 167–171.

Received March 30, 1993, *and in revised form August* 26, 1994, *and May* 1, 1997.