Shortest Point-Visible Paths on Polyhedral Surfaces

Ramtin Khosravi, Mohammad Ghodsi, and Mana Taghdiri

Department of Computer Engineering
Sharif University of Technology
1 {raamtin,ghodsi,taghdiri}@ce.sharif.ac.ir

Abstract. In this paper we study the problem of finding the shortest path between two points lying on the surface of a (possibly nonconvex) polyhedron which is constrained so that a given point on the surface of the polyhedron must be seen from some point in the path. Our algorithm runs in time $O(n^2 \log n)$ for the convex polyhedron and $O(n^3 \log n)$ for nonconvex case. The method used is based on the subdivision of the inward layout and intersecting it with the visibility map of the point to be seen.

1 Introduction

The problem of finding the shortest path between two points lying on the surface of a 3D polyhedron is a basic problem in computational geometry and has many applications in different areas such as motion planning and navigation. Several forms of the problem can be defined as we change the properties of the polyhedron (e.g. considering faces to have weights, being nonconvex, etc.), or constrain the path with different restrictions. An example of the constrained versions is the problem of finding paths which does not go above some given height as studied in [1].

The problem of finding the shortest path between two points on the surface of a polyhedron is well studied in [7], [5], [2], [3]. Especially [2] presents a method for building a subdivision of the surface which can be used for finding the shortest paths from a fixed source to a given query point efficiently. The subdivision can be built in $O(n^2)$ time and the shortest path can be determined in $O(k)$ where k is the complexity of the path itself. The best known algorithm for finding shortest paths on polyhedral surfaces is [3] which finds the shortest path in $O(n \log^2 n)$ using *wavefront propagation* method. However as it does not build any subdivision of the surface, it cannot help us for the method used in this paper.

In this paper we study a variation of the shortest path problem with visibility constraints, which has interesting applications. The problem is to find the shortest path from a source point to a target such that a given point on the surface can be seen from at least one point of the path. The method used here is to find a point on the boundary of the visible image of the point to be seen which has minimum total distance from the source and the target. To the best of our knowledge, this is the first result on this problem.

As the complexity of the visible image of a point on a (possibly nonconvex) polyhedron is quadratic in size of the polyhedron and the algorithms for finding the visibility map are superquadratic in general [6], we assume the visible image of the point to be seen is determined through a preprocessing stage and focus on the problem of finding the shortest path touching the visible image. The running time of our algorithm is $O(n^2 \log n)$ for convex and $O(n^3 \log n)$ for nonconvex polyhedra.

2 Preliminaries

Let P be a closed (possibly nonconvex) polyhedron in 3-space. We denote the surface of P by ∂P. By *interior* of P we mean the portion of the space enclosed by ∂P not containing ∂P itself. We consider ∂P to be specified by a set of faces, edges and vertices. Without loss of generality we assume that all faces are triangles. We are given three special points on the surface, namely p, s, and t. The problem is to find the shortest path π between s and t lying on ∂P, from which the point p is visible. The point p is *visible* from π if there is at least one point q on π through which p is visible, i.e. the straight line segment joining p and q does not intersect the interior of P.

2.1 Shortest Paths on a Polyhedron

We borrow the terminology from [5]. Two faces f and f' are said to be *edge-adjacent* if they share a common edge e. A *sequence of edge-adjacent faces* is a list of one or more faces $\mathcal{F} = (f_1, f_2, \ldots, f_{k+1})$ such that face f_i is edge-adjacent to face f_{i+1} (sharing common edge e_i). We refer to the (possibly empty) list of edges $\mathcal{E} = (e_1, e_2, \ldots, e_k)$ as an *edge sequence* and to the vertex of face f_1 which is opposite e_1 as the *root* of $\mathcal{E}$.

Each face has a two dimensional coordinate system associated with it. If faces f and f' are edge-adjacent sharing edge e, we define the *planar unfolding* of face f' onto face f as the image of points of f' when rotated about the line through e into the plane of f such that the points of f' fall on the *opposite* side of e to points of f. Extending this notation, we say that we *unfold* an edge sequence $\mathcal{E} = (e_1, e_2, \ldots, e_k)$ as follows: Rotate f_1 around e_1 until its plane coincides with that of f_2, rotate f_1 and f_2 around e_2 until their plane coincides with that of f_3, continue in this way until all faces $(f_1, f_2, \ldots, f_k)$ lie in the plane of f_{k+1}.

A path on ∂P is called *geodesic* if it is locally optimal and cannot be shortened by small perturbations. We say a path π *connects* edge sequence $\mathcal{E} = (e_1, e_2, \ldots, e_k)$ if π consists of segments which join interior points of $e_1, e_2, \ldots, e_k$ (in that order). Figure 1 shows a shortest path π from s to t unfolded along the edge sequence connected by it $(e_1, e_2, \ldots, e_k)$.

The following lemmas express some facts about geodesic paths:

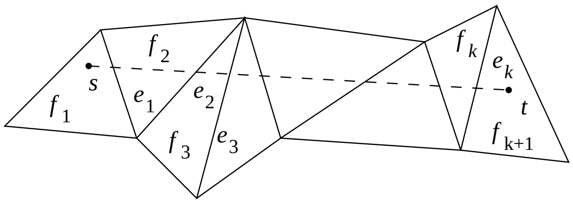

Fig. 1. Shortest path from s to t unfolded along its edge sequence

Lemma 1 (Mitchell, Mount and Papadimitriou [5]). *If π is a geodesic path which connects the edge sequence $\mathcal{E}$, then the planar unfolding of π along $\mathcal{E}$ is a straight line segment.*

Lemma 2 (Sharir and Schorr [7]). *If P is a convex polyhedron then the shortest path between any two points on ∂P cannot pass through a vertex of P except at its end-points.*

Lemma 3 (Mitchell, Mount and Papadimitriou [5]). *The general form of a geodesic path is a path which goes through an alternating sequence of vertices and (possibly empty) edge sequences such that the unfolded image of the path along any edge sequence is a straight line segment and the angle of the path passing through a vertex is greater than or equal to π. The general form of an optimal path is the same as that of a geodesic path, except that no edge can appear in more than one edge sequence and each edge sequence must be simple*

Lemma 4 (Mitchell, Mount and Papadimitriou [5]). *If $\pi(x)$ and $\pi(y)$ are optimal paths from s to points x and y, then they can intersect only at vertices of P, and if they do at v, then the subpath of $\pi(x)$ from s to v has the same length as that subpath of $\pi(y)$ from s to v.*

2.2 Point-visible Paths

The *visibility map* of a point p, denoted by $\mathcal{R}_p$, is the subset of ∂P from which p is visible. In the case of convex polyhedra, $\mathcal{R}_p$ is a connected region whose boundary consists of $O(n)$ edges of P. In nonconvex case, $\mathcal{R}_p$ is a set of $O(n)$ connected regions. The boundary of each region consists of $O(n)$ line segments (not necessarily edges of P). Thus the total complexity of $\mathcal{R}_p$ is $O(n^2)$ in this case. We define $\mathcal{B}_p$ to be the boundary of $\mathcal{R}_p$.

With the definitions above we can say a point p is *visible* from a path π if the intersection of π and $\mathcal{R}_p$ is non-empty. This intersection may be at one point or one or more line segments. We say π is p*-visible* if p is visible from it.

Let π be the shortest p-visible path between s and t. The intersection of π and $\mathcal{B}_p$ is a non-empty set of points such as Q. At least one point in Q such as q has the property that the two subpaths of π from s to q and from q to t are optimal. For if not, we can replace the two subpaths with the optimal ones, resulting a shorter p-visible path from s to t (Fig. 2). From the property stated above, we conclude that the image of each subpath after unfolding the corresponding edge sequence is a straight line segment.

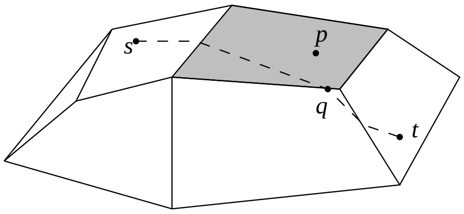

Fig. 2. Shortest p-visible path between s and t. The shaded area is $\mathcal{R}_p$.

3 Computing Shortest p-Visible paths

Consider a maximal set of points on $\mathcal{B}_p$ whose shortest paths to a point x connect the same edge sequence. As the following lemma states, such a set forms a connected interval on $\mathcal{B}_p$, which we call it a *linearity interval with respect to x* or simply an *L-interval* with respect to x (Fig. 3). Clearly the set of all L-intervals with respect to a point x is a partition of $\mathcal{B}_p$.

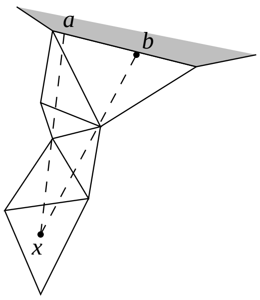

Fig. 3. The segment $\overline{ab}$ is an L-interval with respect to x. The shaded area is part of $\mathcal{R}_p$.

Lemma 1. *A maximal set of points on one edge of $\mathcal{B}_p$ whose shortest paths to a point x connect the same edge sequence is a connected interval.*

Proof. Suppose I be such a set with the edge sequence $\mathcal{E} = (e_1, e_2, \ldots, e_k)$ and I_1 and I_2 be two disjoint connected subsets of I (Fig. 4). Consider a point q between points of I_1 and I_2. Since I is maximal, q has a different shortest edge sequence than that of I_1 and I_2, say $\mathcal{E}' = (e'_1, e'_2, \ldots, e'_p)$. Consider the longest common subsequences of $\mathcal{E}$ and $\mathcal{E}'$ from the end of the sequences. As I_1 and I_2 lie on the same edge of $\mathcal{B}_p$, so does q, thus the common subsequence mentioned above is non-empty. Let e be the first edge in this subsequence and f be the face containing e which is visited first in the face sequence from source to points of I and q. Let e_1 and e_2 be the edges of f other than e. Since $\mathcal{E}$ and $\mathcal{E}'$ differ in edges prior to e, either $e_1 \in \mathcal{E}$ and $e_2 \in \mathcal{E}'$ or converse, i.e. shortest paths to I cross one of the two edges and the shortest path to q crosses the other. Since I_1 and I_2 are on both sides of q, and the unfolded image of shortest paths is a line segment, the shortest path to q must intersect the shortest paths to either I_1 or I_2 and this contradicts lemma 4. □

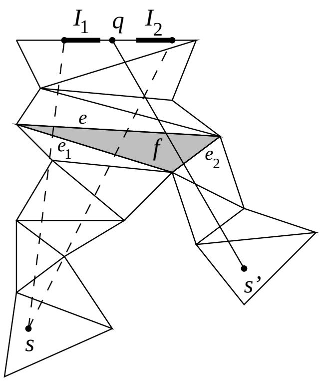

Fig. 4. Proof of the lemma 1.

Suppose L_x and L_y be the set of L-intervals with respect to points x and y. The intervals obtained from overlapping intervals of L_x and L_y are called *L-intervals with respect to the pair x and y*. We name the set of such intervals $L_{x,y}$ (Fig. 5). Clearly $L_{x,y}$ partitions $\mathcal{B}_p$.

Lemma 2. *Let I be an L-interval with respect to the pair x and y. One can find the point $q(I)$ on I with minimum total distance to x and y in constant time.*

Proof. I has the property that the two edge sequences connected by the shortest paths from a point on I to x and y are the same for all points of I. From this property we can find the point $q(I)$ on I whose total distance to x and y is minimum in constant time using elementary techniques of analytical geometry. We omit the details of the computation here. □

Based on the preceding lemma, the shortest p-visible path between s and t can be found by performing the following steps:

1. Compute the set of intervals $L_{s,t}$ on $\mathcal{B}_p$

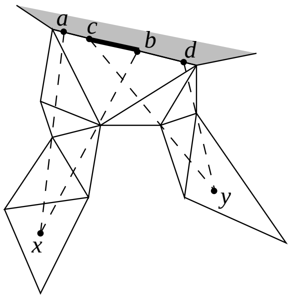

Fig. 5. $\overline{cb}$ is an L-interval with respect to the pair x and y.

2. For each interval $I \in L_{s,t}$ find the point $q(I)$ which has the minimum total distance from s and t.
3. Let q be the point with the minimum total distance from s and t among $\{q(I) : I \in L_{s,t}\}$.
4. The shortest p-visible path between s and t is the shortest path between s and q followed by the shortest path between q and t.

In order to compute the set of intervals $L_{s,t}$ in step 1 above, we compute L_s and L_t separately, then overlapping the intervals to obtain $L_{s,t}$ is straightforward.

3.1 Computing *L*-intervals

For computing the set of L-intervals with respect to a point x on the surface we use a subdivision presented in [2]. Below we review the method for building the subdivision.

Given a source point x on ∂P, one can construct a *shortest path tree* for a given polyhedron in $O(n^2)$ time and $O(n)$ space using Chen and Han's algorithm [2]. This structure holds the information about the shortest paths from x to vertices of P in such a way that for any vertex v of P, the edge sequence connected by the shortest path from s to v can be determined in time proportional to the size of the edge sequence.

One can cut the surface of P along the shortest paths from x to all of the vertices of P. The resulting surface can be laid out on a common plane. The layout obtained in this manner is called the *inward layout* of P and is a star-shaped polygon. The vertices of this polygon are the vertices of P and *images of the source point* x under different unfoldings and the edges are shortest paths from the source to the vertices of P.

A subdivision of the inward layout can be obtained by constructing the Voronoi diagram on the layout with respect to the images of the source point (Fig. 6). This subdivision has the property that The points in the same region are closer to the corresponding image of the source than to other images, and their shortest paths from the source pass the same edge sequence. As the number of images are $O(n)$, the subdivision can be built in $O(n \log n)$ using known algorithms for computing Voronoi diagrams. For the nonconvex case, the inward layout may overlap itself. The algorithm for computing Voronoi diagram in this case is slightly different and is given in [2].

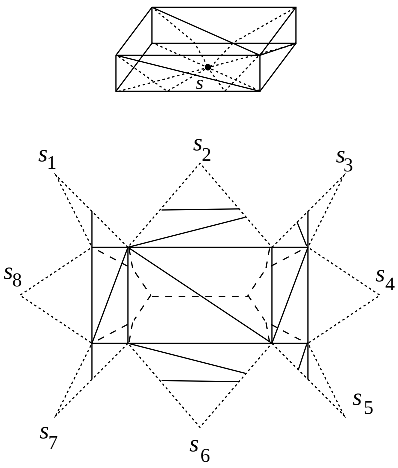

Fig. 6. The inward layout of a box. The edges of the subdivision are dashed lines.

It is clear that the L-intervals on $\mathcal{B}_p$ with respect to point x are the intervals formed by intersection of $\mathcal{B}_p$ and the subdivision mentioned above considering x as the source point.

3.2 The Nonconvex Case

In the nonconvex case, the shortest path between two points may pass through vertices of the polyhedron. Suppose v be the last vertex on the shortest path π from x to y. π consists of two optimal subpaths: one from x to v and one from v to y, while the latter can be characterized by the edge sequence connected with it. We may view each vertex v as a *pseudo-source* with an initial distance equal to the length of the shortest path from the source to it. This causes the shortest path tree to differ slightly from the convex case by introducing some *vertex nodes* in it which correspond to pseudo-sources. However the time and space needed to construct the tree remains the same. In this case the subdivision of the surface is obtained by constructing the Voronoi diagram on the inward layout of the polyhedron with respect to the images of the source and pseudo-sources. For a point in the region corresponding to one pseudo-source, the shortest path is obtained by first finding the shortest path from source to the vertex associated by the pseudo-source, then following the edge sequence specified by the subdivision region.

In this case an L-interval on $\mathcal{B}_p$ with respect to x has the property that there is a vertex v of the polyhedron which every shortest path from x to a point on the interval passes through v as the last vertex, and the edge sequence from v to points on the interval is the same. As the first part of the path is fixed among the points on interval, given an L-interval I with respect to s and t we can still find the point $q(I)$ whose total distance from s and t is minimum in constant time provided that during computing intervals we store the distance to pseudo-source associated with the interval.

3.3 Analysis of the Algorithm

We consider two stages for the algorithm: computing L-intervals with respect to the pair s and t, and finding the shortest p-visible path. The first stage is analyzed by the following two lemmas.

Lemma 3. *The number of L-intervals on $\mathcal{B}_p$ with respect to a point is $O(n^2C(\mathcal{B}_p))$ where $C(\mathcal{B}_p)$ is the number of connected components of $\mathcal{B}_p$.*

Proof. From lemma 1 we know the intersection of any edge sequence corresponding to a shortest path with each edge of $\mathcal{B}_p$ is at most one L-interval. From this, we can conclude that the total number of intervals on each edge of $\mathcal{B}_p$ is $O(n)$. As in general, each connected component of $\mathcal{B}_p$ has $O(n)$ edges, the total number of L-intervals on $\mathcal{B}_p$ is $O(n^2C(\mathcal{B}_p))$. □

Thus in worst case, there are $O(n^2)$ and $O(n^3)$ L-intervals on $\mathcal{B}_p$ for convex and nonconvex polyhedra respectively. It can be shown that the bound in lemma 3 is tight. As an example, in Fig. 7(b), $\mathcal{R}_p$ consists of all faces adjacent to the top vertex p, and the subdivision with respect to x intersects $\mathcal{B}_p$ in $O(n^2)$ edges if the number of faces added to the diamond polyhedron (Fig. 7(a)) be $O(n)$.

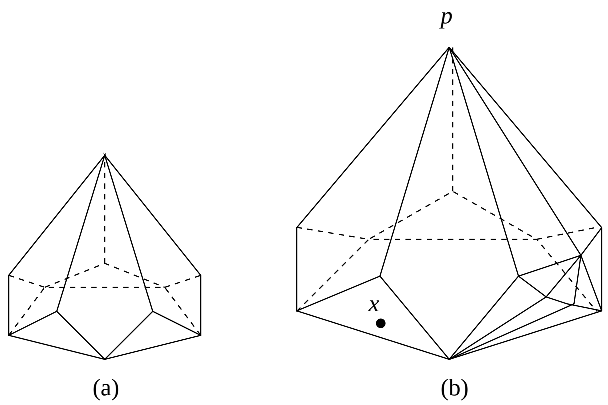

Fig. 7. (a) A diamond polyhedron which have $O(n)$ *sides*
(b) Diamond polyhedron with a number of faces added to it.

Lemma 4. *Computing L-intervals on $\mathcal{B}_p$ with respect to the pair s and t can be done in $O(n^2 \log n)$ for convex and $O(n^3 \log n)$ for nonconvex polyhedra.*

Proof. Computing the set of L-intervals with respect to a point x requires two steps:

1. Computing the inward layout and building the subdivision
2. Computing the intervals formed by intersecting $\mathcal{B}_p$ with the subdivision.

The first step can be done in $O(n^2)$ time in both convex and nonconvex cases while the resulting subdivision has $O(n)$ regions [2]. For computing the intervals we consider the two cases separately:
Convex case. In this case $\mathcal{B}_p$ consists of $O(n)$ segments so $O(n^2 \log n)$ time is sufficient for the computation of L intervals with respect to x using known map overlay algorithms. It can be shown that there are polyhedra for which this time bound is necessary.
Nonconvex case. In this case $\mathcal{B}_p$ consists of $O(n^2)$ segments so $O(n^3 \log n)$ time is needed for computing intervals. Thus the total time required for computing the set of L-intervals with respect to a point x is In $O(n^2 \log n)$ in convex case and $O(n^3 \log n)$ in nonconvex case.

After computing L-intervals with respect to s and t, we can find the overlapping intervals proportional to the number of intervals with respect to s and with respect to t. From the lemma 3, overlapping can be done in $O(n^2)$ for convex case and in $O(n^3)$ for nonconvex case. □

Theorem 1. *The time required for computing shortest p-visible path between two points s and t is $O(n^2 \log n)$ for convex and $O(n^3 \log n)$ for nonconvex polyhedra.*

Proof. From the lemma 4, one can find the L-intervals on $\mathcal{B}_p$ in $O(n^2 \log n)$ for convex and $O(n^3 \log n)$ for nonconvex polyhedra. From lemma 2 we can find in constant time, the point $q(I)$ on each L-interval I whose total distance from s and t is minimum. As the number of intervals is $O(nC(\mathcal{B}_p))$, finding the point q with minimum total distance from s and t is possible in $O(n^2 C(\mathcal{B}_p))$. which is asymptotically less than the time required for finding the intervals in both cases. After finding point q, one can easily construct the shortest p-visible path between s and t in time $O(k)$ where k is the number of edges connected by the path (which is $O(n)$). Thus the total time required by the algorithm is $O(n^2 \log n)$ for convex and $O(n^3 \log n)$ for nonconvex cases. □

4 Conclusions

We proposed an algorithm for finding the shortest path between two points on general polyhedra from which a given point is visible. The running time of our algorithm is $O(n^2 \log n)$ for convex and $O(n^3 \log n)$ for nonconvex polyhedra. The algorithm uses the inward layout and the subdivision based on it. Extending the algorithm to the case in which we want to see multiple points from the path is not easy since it introduces multiple regions (neighborhoods) which must be touched by the shortest path. The problem is similar to *TSP with neighborhoods* which is studied in [4]. Since the problem is NP-hard, finding an approximation algorithm seems to be the most feasible approach for solving it.

References

1. M. de Berg and M. van Kreveld, Trekking in the Alps Without Freezing or Getting Tired, *Algorithmica*, 18:306-323, 1997.
2. J. Chen and Y. Han, Shortest Paths on a Polyhedron, *Iternat. J. Comput. Geom. Appl.*, 6:127-144, 1996.
3. S. Kapoor, Efficient Computation of Geodesic Shortest Paths, In *Proc. 32nd Annu. ACM Sympos. Theory Comput.*, 1999.
4. C. S. Mata and J. S. B. Mitchell, Approximation Algorithms for Geometric Tour and Network Design Problems, In *Proc. 11th Annu. Sympos. Comput. Geom.*, pp. 360-369, 1995.
5. J. S. B. Mitchell, D. M. Mount and C. H. Papadimitriou, The Discrete Geodesic Problem, *SIAM J. Comput.*, 16:647-668, 1987.
6. J. O'Rourke, Visibility, In J. E. Goodman and J. O'Rourke, eds., *Handbook of Discrite and Computational Geometry*, chapter 25, pp. 467-480, CRC Press LLC, Boca Raton, FL, 1995.
7. M. Sharir and A. Schorr, Shortest Paths in Polyhedral Spaces, *SIAM J. Comput.*, 15:193-215, 1986.