Speed-Constrained Geodesic Fréchet Distance Inside a Simple Polygon[*]

Anil Maheshwari Jörg-Rüdiger Sack Kaveh Shahbaz Hamid Zarrabi-Zadeh

Abstract

Given two polygonal curves inside a simple polygon, we study the problem of finding the Fréchet distance between the two curves under the following two conditions (i) the distance between two points on the curves is measured as the length of the shortest path between them lying inside the simple polygon, and (ii) the traversal along each segment of the polygonal curves is restricted to be between a minimum and a maximum permissible speed assigned to that segment.

1 Introduction

Fréchet distance is a widely-used metric for measuring the similarity between two geometric objects. It has applications in morphing, protein structure alignment, handwriting recognition, GIS, etc. This measure is often interpreted as the minimum-length leash needed for a person to walk a dog, while each of them is traversing a pre-specified polygonal curve.

Various variants of the Fréchet distance have been studied in the literature. Cook and Wenk [2] studied the geodesic Fréchet distance inside a simple polygon. In this variant, the leash is constrained to the interior of a simple polygon. Therefore, a *geodesic* distance is used to measure the length of the leash, which is the length of the shortest path inside the polygon connecting the two endpoints of the leash. In [2], it is shown that the geodesic Fréchet distance between two polygonal curves of size n inside a simple polygon of size k can be computed in $O(n^2 \log kn \log n + k)$ expected time and $O(n^2 + k)$ space.

In [3], Maheshwari *et al.* introduced a generalization of the Fréchet distance, in which users are allowed to set speed limits on each segment. They showed that for two polygonal curves of size n with speed limits assigned to their segments, the speed-constrained Fréchet distance can be computed in $O(n^2 \log^2 n)$ time and $O(n^2)$ space. Note that in this variant, there is no restriction to stay inside a simple polygon and thus, leash lengths are measured with Euclidean distance.

In this paper, we study the speed-constrained geodesic Fréchet distance inside a simple polygon which is a simultaneous generalization of both Fréchet distances studied in [2] and [3]. The decision version of the problem is formulated as follows: Let P and Q be two polygonal curves inside a simple polygon, with minimum and maximum permissible speeds assigned to each segment of P and Q. For a given $\varepsilon \geqslant 0$, can two point objects traverse P and Q with permissible speeds (without backtracking) and, throughout the entire traversal, remain at geodesic distance at most ε from each other? The objective in the optimization problem is to find the smallest such ε.

In this paper, we show that the decision version of the speed-constrained geodesic Fréchet distance problem can be solved in $O(n^2(k + n))$ time and $O(n^2 + k)$ space, where n is the number of segments in the curves, and k is the complexity of the simple polygon. Combined with a standard parametric search technique, this leads to a solution to the optimization problem in $O(n^2(k + n) \log n)$ time and $O(n^2 + k)$ space.

Algorithms for computing various variants of the Fréchet distance are typically based on computing a free space diagram consisting of $O(n^2)$ cells, similar to the one used in [1], and then propagating the reachability information one by one through the cells. While we adopt this general approach, the construction of the free space diagram is more challenging in our problem as we need to compute the whole free space inside each cell. This is in contrast to other variants that only need to compute the free space on the boundaries of the cells. A main contribution of this paper is thus to fully describe the structure of the free space inside a cell, establish its complexity, and show how it can be computed efficiently. Propagating the reachability information through the cells is also more challenging in our problem compared to the previous ones in [2, 3], as here, the shape of the free space inside a cell can substantially affect the projection of the reachable intervals on its boundaries.

2 Preliminaries

A *polygonal curve* in $\mathbb{R}^d$ is a continuous function $P : [0, n] \rightarrow \mathbb{R}^d$ with $n \in \mathbb{N}$, such that for each $i \in \{0, \ldots, n-1\}$, the restriction of P to the interval $[i, i+1]$ is affine (i.e., forms a line segment). The

[*]Research supported by NSERC and SUN Microsystems. Authors' affiliation: School of Computer Science, Carleton University, Ottawa, Ontario K1S 5B6, Canada. Email: {anil,sack,kshahbaz,zarrabi}@scs.carleton.ca.

integer n is called the *length* of P. Moreover, the sequence $P(0), \dots, P(n)$ represents the set of *vertices* of P. For each $i \in \{1, \dots, n\}$, we denote the line segment $P(i-1)P(i)$ by P_i. Given a simple polygon K and two points $p, q \in K$, the *geodesic distance* of p and q with respect to K, denoted by $d_K(p,q)$, is defined as the length of the shortest path between p and q that lies completely inside K.

Speed-constrained geodesic Fréchet distance. Let P be a polygonal curve such that assigned to each segment S of P, there is a pair of non-negative real numbers $(v_{\min}(S), v_{\max}(S))$ specifying the minimum and the maximum permissible speed for moving along S. We define a *speed-constrained parametrization of* P to be a continuous surjective function $f : [0,T] \to [0,n]$ with $T > 0$ such that for any $i \in \{1, \dots, n\}$, the slope of f at all points $t \in [f^{-1}(i-1), f^{-1}(i)]$ is within $[\bar{v}_{\min}(P_i), \bar{v}_{\max}(P_i)]$, where $\bar{v}_{\min}(S) = v_{\min}(S)/\|S\|$ and $\bar{v}_{\max}(S) = v_{\max}(S)/\|S\|$.

Given a simple polygon K and two polygonal curves P and Q inside K of lengths n and m respectively with speed limits assigned to their segments, the *speed-constrained geodesic Fréchet distance* of P and Q inside K is defined as

$$\delta_{\hat{F}}(P,Q) = \inf_{\alpha,\beta} \max_{t \in [0,T]} d_K(P(\alpha(t)), Q(\beta(t))),$$

where $\alpha : [0,T] \to [0,n]$ ranges over all speed-constrained parameterizations of P and $\beta : [0,T] \to [0,m]$ ranges over all speed-constrained parameterizations of Q.

Free space diagram. Let $\mathcal{G}_{n\times m} = [0,n] \times [0,m]$ be a n by m rectangle in the plane. Each point $(s,t) \in \mathcal{G}_{n\times m}$ uniquely represents a pair of points $(P(s), Q(t))$ on the polygonal curves P and Q. We decompose $\mathcal{G}_{n\times m}$ into $n \times m$ unit grid cells $\mathcal{C}_{ij} = [i-1,i] \times [j-1,j]$ for $(i,j) \in \{1,\dots,n\} \times \{1,\dots,m\}$, where each cell $\mathcal{C}_{ij}$ corresponds to a segment P_i on P and a segment Q_j on Q. Given two polygonal curves P and Q inside a simple polygon K and a parameter $\varepsilon \geqslant 0$, the *free space* $\mathcal{F}_\varepsilon$ is defined as $\mathcal{F}_\varepsilon = \{(s,t) \in \mathcal{G}_{n\times m} \mid d_K(P(s), Q(t)) \leqslant \varepsilon\}$. We denote by L_{ij} (resp., by B_{ij}) the left (resp., bottom) line segment bounding $\mathcal{C}_{ij}$. The *entry side* of $\mathcal{C}_{ij}$ is defined as $\text{entry}(\mathcal{C}_{ij}) = L_{ij} \cup B_{ij}$, and its *exit side* as $\text{exit}(\mathcal{C}_{ij}) = B_{i,j+1} \cup L_{i+1,j}$. Given two points p and q on the boundary of a cell, we say that p is *before* q, denoted by $p \prec q$, if either $p_x < q_x$ or $p_x = q_x$ & $p_y > q_y$.

Hourglass data structure. Fix a simple polygon K. Given two points $p, q \in K$, we denote by $\pi(p,q)$ the shortest path between p and q that lies inside K, and denote its length by $\|\pi(p,q)\|$. Let $\overline{ab}$ and $\overline{cd}$ be two non-crossing line segments inside K. The *hourglass* $\mathcal{H}_{\overline{ab},\overline{cd}}$ is defined as the maximal region bounded by the segments $\overline{ab}$ and $\overline{cd}$, and the shortest path chains $\pi(a,c)$, $\pi(a,d)$, $\pi(b,c)$ and $\pi(b,d)$. Three examples of hourglasses are illustrated in Figure 1. Note that for any two points $p \in \overline{ab}$ and $q \in \overline{cd}$, the shortest path $\pi(p,q)$ is contained in $\mathcal{H}_{\overline{ab},\overline{cd}}$. The intersection of $\mathcal{H}_{\overline{ab},\overline{cd}}$ and the boundary of K consists of at most four polygonal curves, each of which is called a *chain* of $\mathcal{H}_{\overline{ab},\overline{cd}}$.

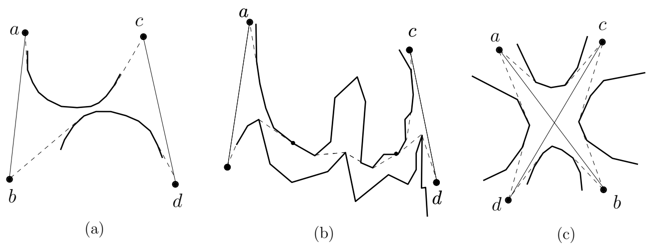

Figure 1: (a) An open hourglass (b) A closed hourglass (c) An intersecting hourglass.

3 Computing the Free Space Inside a Cell

In the classical Fréchet distance problem [1], the free space inside each cell is convex and can be determined in $O(1)$ time. When distances are geodesic, the free space is not necessarily convex, but it is still connected and xy-monotone [2]. Therefore, to solve the geodesic Fréchet distance (without speed limits), one only needs to compute the free space on the boundaries of the cells as in [2]. In contrast, in our generalized version we need to compute the full description of the free space in the interior of the cells as well in order to propagate the reachability information correctly.

We use the hourglass data structure to compute the boundary of the free space inside a cell. Consider an hourglass $\mathcal{H}_{\overline{ab},\overline{cd}}$ and two points $p \in \overline{ab}$ and $q \in \overline{cd}$. The shortest path $\pi(p,q)$ is either a straight segment (in case p and q see each other), or consists of two tangents from p and q to the chains of $\mathcal{H}_{\overline{ab},\overline{cd}}$ plus a subpath between the two tangent points. We denote this subpath by $\sigma(p,q)$. Note that $\sigma(p,q)$ consists of a sequence of vertices of K, lying on at most two chains of the hourglass.

Definition 1 *Consider an hourglass $\mathcal{H}_{\overline{ab},\overline{cd}}$ and two intervals $\overline{a'b'} \subseteq \overline{ab}$ and $\overline{c'd'} \subseteq \overline{cd}$, so that for any $p \in \overline{a'b'}$ and any $q \in \overline{c'd'}$, $\sigma(p,q)$ is the same. The region bounded by the intervals $\overline{a'b'}$ and $\overline{c'd'}$ and the paths $\pi(a',c')$ and $\pi(b',d')$ is called a butterfly, and is denoted by $\mathcal{B}_{\overline{a'b'},\overline{c'd'}}$ (see Figure 2).*

Lemma 1 *Given a butterfly $\mathcal{B}_{\overline{a'b'},\overline{c'd'}}$, the function $f(p,q) = \|\pi(p,q)\|$ over the domain $[a',b'] \times [c',d']$ is a hyperbolic surface.*

Proof. Fix a point $p \in \overline{a'b'}$ and a point $q \in \overline{c'd'}$. Let k_1 and k_2 be the two endpoints of $\sigma(p,q)$. Then $\|\pi(p,q)\| = \|pk_1\| + \|\sigma(p,q)\| + \|k_2q\|$. By the butterfly property, k_1, k_2, and $\|\sigma(p,q)\|$ are fixed for all p and q in the domain. Therefore, $\|\pi(p,q)\|$ is the sum of two L_2 distances plus a constant, which forms a hyperbolic surface. □

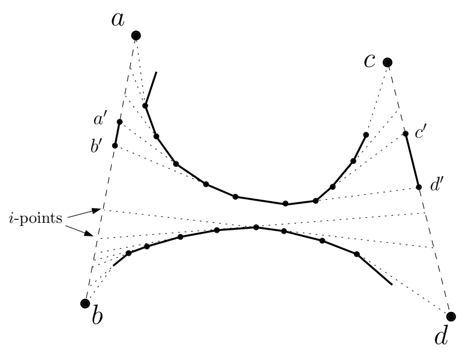

Figure 2: An hourglass $\mathcal{H}_{\overline{ab},\overline{cd}}$ with a butterfly $\mathcal{B}_{\overline{a'b'},\overline{c'd'}}$.

Consider an edge e on a chain of the hourglass $\mathcal{H}_{\overline{ab},\overline{cd}}$. Extend e to a line and find its intersection with $\overline{ab}$ and $\overline{cd}$ (as shown in Figure 2). We call such an intersection point an *i-point*. Note that the number of i-points on each of the segments $\overline{ab}$ and $\overline{cd}$ is $O(k)$.

Observation 1 *Any two consecutive i-points $i_1, i_2 \in \overline{ab}$ and any two consecutive i-points $j_1, j_2 \in \overline{cd}$ form a butterfly $\mathcal{B}_{\overline{i_1i_2},\overline{j_1j_2}}$.*

Consider two polygonal curves P and Q inside K. Let $P_i = \overline{ab}$ be a segment of P, and $Q_j = \overline{cd}$ be a segment of Q. By dividing $\overline{ab}$ and $\overline{cd}$ at i-points, the corresponding cell $\mathcal{C}_{ij}$ in the free space diagram is decomposed into $O(k^2)$ subcells, where each subcell corresponds to a butterfly (see Figure 3).

Let $f(p,q) = \|\pi(p,q)\|$ be a function defined over all $(p,q) \in [a,b] \times [c,d]$. The intersection of the plane $z = \varepsilon$ with the function f determines the boundary of $\mathcal{F}_\varepsilon$ inside the cell $\mathcal{C}_{ij}$. The boundary of $\mathcal{F}_\varepsilon$ crosses the boundary of each subcell in at most two points, each of which is called a *c-point*. The following two lemmas describe the structure of the free space inside $\mathcal{C}_{ij}$.

Lemma 2 *Any two consecutive c-points on the boundary of $\mathcal{F}_\varepsilon$ are connected with a hyperbolic arc, and the line segment connecting the two endpoints of the arc lies completely inside $\mathcal{F}_\varepsilon$.*

Proof. This is a corollary of Lemma 1. □

Lemma 3 *The number of c-points inside a cell is $O(k)$.*

Proof. This follows from the fact that any xy-monotone curve intersecting an $n \times m$ (non-uniform) grid can cross at most $2(n+m)$ cells of the grid. □

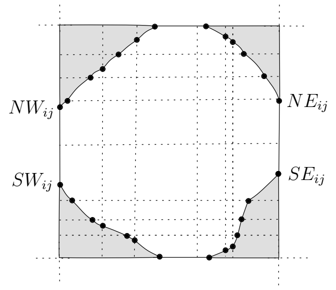

Figure 3: The free space inside a cell.

Computing c-points. Our algorithm for computing c-points is based on the following observation.

Observation 2 *Consider an hourglass $\mathcal{H}_{\overline{ab},\overline{cd}}$ and a fixed $\varepsilon > 0$. Let p be a point moving on $\overline{ab}$, and let q be a point that moves on the line $\overleftrightarrow{cd}$ to maintain geodesic distance ε from p. When p moves monotonically from a to b, q has at most one directional change along $\overleftrightarrow{cd}$.*

Observation 2 enables us to compute all c-points inside a cell by two linear walks. Details are provided in Algorithm 1. In this algorithm, h_c refers to a point on $\overline{ab}$ which is closest to c, $p_1 \prec_{\overrightarrow{ab}} p_2$ means that p_1 is before p_2 in direction $\overrightarrow{ab}$, and $F(p,q)$ refers to the unique point in the free space diagram corresponding to a point $p \in P$ and $q \in Q$. The output of the algorithm is four connected c-point chains as depicted in Figure 3.

Algorithm 1 COMPUTING C-POINTS INSIDE A CELL

```
Input: An hourglass $\mathcal{H}_{\overline{ab},\overline{cd}}$ corresponding to a cell $\mathcal{C}_{ij}$
    and a fixed $\varepsilon > 0$.
1: Compute $i$-points on $\overline{ab}$ and $\overline{cd}$.
2: Find $q_1, q_2 \in \overleftrightarrow{cd}$ s.t. $\|\pi(a,q_1)\| = \|\pi(a,q_2)\| = \varepsilon$.
3: Set $\eta_1 = q_1$ and $\eta_2 = q_2$, assuming that $q_1 \prec_{\overrightarrow{cd}} q_2$.
4: Set $\mu = a$.
5: while $\mu$ has not reached $b$ do
6:     Move $\mu$ in direction $\overrightarrow{ab}$, and move $\eta_1$ on $\overleftrightarrow{cd}$ s.t.
       $\|\pi(\mu,\eta_1)\| = \varepsilon$ until either $\mu$ or $\eta_1$ reaches an
       $i$-point.
7:     if $F(\mu,\eta_1) \in \mathcal{C}_{ij}$ then
8:         Insert $F(\mu,\eta_1)$ into $SW_{ij}$ if $\mu \prec_{\overrightarrow{ab}} h_c$, other-
           wise insert $F(\mu,\eta_1)$ into $NW_{ij}$.
9: Repeat lines 4–8 with $\eta_2$ instead of $\eta_1$ to obtain
   $NE_{ij}$ and $SE_{ij}$.
10: return $NW_{ij}$, $SW_{ij}$, $NE_{ij}$, and $SE_{ij}$.
```

Theorem 4 *Algorithm 1 computes the free space inside a cell in $O(k)$ time.*

4 The Decision Problem

In this section, we show how the decision version of our Fréchet distance problem can be solved efficiently. We use the notation of [3]. A path $\mathcal{P} \subset \mathcal{G}_{n\times m}$ is called *slope-constrained* if for any point $(s,t) \in \mathcal{P} \cap \mathcal{C}_{ij}$, the slope of $\mathcal{P}$ at (s,t) is within $\text{minSlope}_{ij} = \bar{v}_{\min}(Q_j)/\bar{v}_{\max}(P_i)$ and $\text{maxSlope}_{ij} = \bar{v}_{\max}(Q_j)/\bar{v}_{\min}(P_i)$. A point $(s,t) \in \mathcal{F}_\varepsilon$ is called *reachable* if there is a slope-constrained path from $(0,0)$ to (s,t) in $\mathcal{F}_\varepsilon$. As shown in [3], $\delta_{\hat{F}}(P,Q) \leqslant \varepsilon$ if and only if the point (n,m) is reachable.

Reachable points on the entry side of each cell form a set of $O(n^2)$ disjoint intervals, each of which is called a *reachable interval* [3]. To decide if (n,m) is reachable, the general approach is to propagate the reachability information one by one, in the row-major order, from $\mathcal{C}_{0,0}$ to $\mathcal{C}_{nm}$. The propagation in each cell $\mathcal{C}_{ij}$ involves projecting the set of reachable intervals from the entry side of the cell to its exit side.

Since the free space inside a cell is not necessarily convex in our problem, the projection can be affected by the boundary of $\mathcal{F}_\varepsilon$ inside a cell (see Figure 4). We use the c-point information computed in the previous section to compute projections. Indeed, only c-points on the convex hull of NW_{ij} and SE_{ij} are needed to compute correct projections. Since c-points inside each chain are stored in a sorted x (and y) order, the convex hull of the chains can be computed using a Graham scan in $O(k)$ time. We call the convex hull of NW_{ij} (resp., SE_{ij}) the *left chain* (resp., the *right chain*) of $\mathcal{C}_{ij}$.

Given a point $p \in \text{entry}(\mathcal{C}_{ij})$, Algorithm 2 computes the projection of p onto $\text{exit}(\mathcal{C}_{ij})$ in $O(\log k)$ time.

Algorithm 2 PROJECTION FUNCTION

```
Input: A point $p \in \text{entry}(\mathcal{C}_{ij})$
1: Let $t_\ell$ and $t_r$ be tangents (if they exist) from $p$ to
   the left and to the right chain of $\mathcal{C}_{ij}$, respectively.
2: Let $a_1$ and $a_2$ be the projection of $p$ in directions $t_\ell$
   and $\text{maxSlope}_{ij}$, respectively.
3: Let $b_1$ and $b_2$ be the projection of $p$ in directions $t_r$
   and $\text{minSlope}_{ij}$, respectively.
4: return $[\max(a_1,a_2), \min(b_1,b_2)]$
```

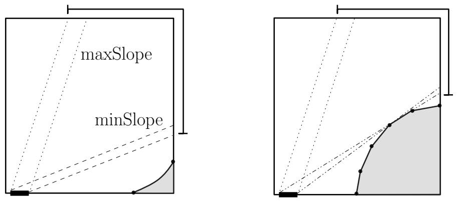

Figure 4: Projecting reachable intervals inside cells with convex and non-convex interior.

Lemma 5 *Given a cell $\mathcal{C}_{ij}$ with r_{ij} reachable intervals on its entry side, we can project all the reachable intervals onto the exit side of $\mathcal{C}_{ij}$ in $O(k + r_{ij})$ time.*

Proof. Let t_1 be a line in direction minSlope_{ij} tangent to the left chain of $\mathcal{C}_{ij}$, and let t_2 be a line in direction maxSlope_{ij} tangent to the right chain of $\mathcal{C}_{ij}$. Let a_1 and a_2 be the intersection points of t_1 and t_2 with $\text{entry}(\mathcal{C}_{ij})$, respectively. For any point $p \in \text{entry}(\mathcal{C}_{ij})$ that lies outside $[a_1, a_2]$, the projection of p is empty. Therefore, we delete those portions of reachable intervals that lie outside $[a_1, a_2]$. Now, the projection of each of the remaining intervals can be simply computed by projecting its two endpoints.

To avoid spending $O(\log k)$ time for projecting each endpoint, we use a cross-ranking technique to reduce the total time needed for computing the tangents in Algorithm 2. Let T_1 be the list of all endpoints of the reachable intervals on $\text{entry}(\mathcal{C}_{ij})$ in $\prec$ order. We construct another list T_2 as follows. Perform an edge traversal of the right chain, starting with the rightmost edge. Each edge encountered is extended to a line until it intersects the entry side at a point which is then added to T_2. We merge T_1 and T_2 (in $\prec$ order) to create a list T. Each item in T has a pointer to its corresponding c-point or reachable interval endpoint, and vice versa. Moreover, each item in T which comes from T_1 keeps a pointer to its preceding item in T which comes from T_2. Now, given a reachable interval endpoint p, to compute the tangent from p to the right chain, we simply find the item $t \in T$ corresponding to p, and then find the item in T_2 preceding t in T. This item uniquely determines the c-point at which the tangent from p to the right chain occurs. We process the left chain in the same way. This enables us to compute each tangent in constant time, after the cross-ranking step, leading to $O(k + r_{ij})$ total time for projecting all endpoints. $\square$

Combined with the fact that $\sum_{0\leqslant i,j\leqslant n} r_{ij} = O(n^3)$ [3], we get the following result:

Theorem 6 *The decision problem can be solved in $O(n^2(k+n))$ time and $O(n^2+k)$ space.*

References

[1] H. Alt and M. Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. of Comput. Geom. Appl.*, 5:75–91, 1995.

[2] A. F. Cook and C. Wenk. Geodesic Fréchet distance inside a simple polygon. In *Proc. 25th STACS*, pages 193–204, 2008.

[3] A. Maheshwari, J.-R. Sack, K. Shahbaz, and H. Zarrabi-Zadeh. Fréchet distance with speed limits. Technical Report TR-10-08, School of Computer Science, Carleton University, 2010. Preliminary version appeared in CCCG 2009.