Staying Close to a Curve*

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

Abstract

Given a point set S and a polygonal curve P in $\mathbb{R}^d$, we study the problem of finding a polygonal curve through S, which has a minimum Fréchet distance to P. We present an efficient algorithm to solve the decision version of this problem in $O(nk^2)$ time, where n and k represent the sizes of P and S, respectively. A curve minimizing the Fréchet distance can be computed in $O(nk^2 \log(nk))$ time. As a by-product, we improve the map matching algorithm of Alt *et al.* by a $\log k$ factor for the case when the map is a complete graph.

1 Introduction

Matching two geometric patterns is a fundamental problem in pattern recognition, protein structure prediction, computer vision, geographic information systems, etc. Usually these patterns consist of line segments and polygonal curves.

One of the most popular ways to measure the similarity of two curves is to use the Fréchet distance. An intuitive way to illustrate the Fréchet distance is as follows. Imagine a person walking his/her dog, where the person and the dog, each travels a pre-specified curve, from beginning to the end, without ever letting go of the leash or backtracking. The Fréchet distance between the two curves is the minimal length of a leash which is necessary. The leash length determines how similar the two curves are to each other: a short leash means the curves are similar, and a long leash means that the curves are different from each other.

Two problem instances naturally arise: decision and optimization. In the *decision problem*, one wants to decide whether two polygonal curves P and Q are within ε Fréchet distance to each other, i.e., if a leash of given length ε suffices. In the *optimization problem*, one wants to determine the minimum such ε. Alt and Godau [2] presented an $O(n^2)$ algorithm for the decision problem, where n denotes the total number of segments in the curves. They also solved the corresponding optimization problem in $O(n^2 \log n)$ time.

In this paper, we address the following variant of the Fréchet distance problem. Given a point set S and a polygonal curve P in $\mathbb{R}^d$ ($d \geqslant 2$), find a polygonal curve Q, with its vertices chosen from S, that minimizes the Fréchet distance to P. Note that the same point of S can be used more than once in Q. In the decision version of the problem, we want to decide if there is polygonal curve Q through S whose Fréchet distance to P is at most ε, for a given $\varepsilon \geqslant 0$. An instance of the decision problem is illustrated in Figure 1.

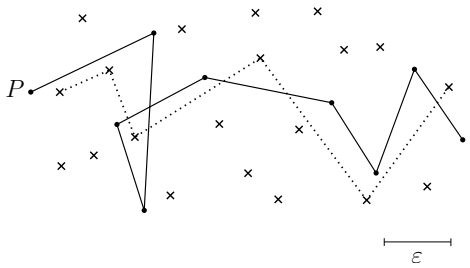

Figure 1: A problem instance.

One can use the map matching algorithm of Alt *et al.* [1] to solve the decision version of this problem by constructing a complete graph G on top of S, and then running Alt *et al.*'s algorithm on G and P. If n and k represent the sizes of P and S, respectively, this leads to a running time of $O(nk^2 \log k)$ for solving the decision problem.

In this paper, we present a simple algorithm to solve the decision version of the above problem in $O(nk^2)$ time. This improves upon the algorithm of Alt *et al.* [1] by a $O(\log k)$ factor for the case when a curve is matched in a complete graph. Our approach is different from and simpler than the approach taken by Alt *et al.* which is a mixture of line sweep, dynamic programming, and Dijkstra's algorithm.

2 Preliminaries

Let $\varepsilon \geqslant 0$ be a real number, and d be a fixed integer. For any point $p \in \mathbb{R}^d$, we define $\mathcal{B}(p, \varepsilon) \equiv \{q \in \mathbb{R}^d : \|pq\| \leqslant \varepsilon\}$ to be a *ball* of radius ε centered at p. Given a line segment $L \subseteq \mathbb{R}^d$, we define $\mathcal{C}(L, \varepsilon) \equiv \cup_{p \in L} \mathcal{B}(p, \varepsilon)$ to be a *cylinder* of radius ε around L (see Figure 2).

A curve in $\mathbb{R}^d$ can be represented as a continuous function $P : [0, 1] \to \mathbb{R}^d$. Given two points $u, v \in P$, we write $u \prec v$, if u is located before v on P. The relation $\preceq$ is defined analogously. For a subcurve $R \subseteq P$, we denote by left(R) and right(R) the first and the last point of R

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

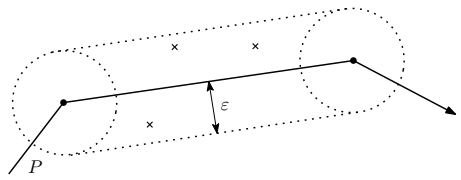

Figure 2: A cylinder of radius ε.

along P, respectively.

Given two curves $\alpha, \beta : [0,1] \to \mathbb{R}^d$, the *Fréchet distance* between α and β is defined as $\delta_F(\alpha,\beta) = \inf_{\sigma,\tau} \max_{t\in[0,1]} \|\alpha(\sigma(t)), \beta(\tau(t))\|$, where $\sigma, \tau : [0,1] \to [0,1]$ range over all continuous non-decreasing surjective functions. The following two observations are immediate.

Observation 1 *Given four points $a, b, c, d \in \mathbb{R}^d$, if $\|ab\| \leqslant \varepsilon$ and $\|cd\| \leqslant \varepsilon$, then $\delta_F(\overrightarrow{ac}, \overrightarrow{bd}) \leqslant \varepsilon$.*

Observation 2 *Let α_1, α_2, β_1, and β_2 be four curves such that $\delta_F(\alpha_1,\beta_1) \leqslant \varepsilon$ and $\delta_F(\alpha_2,\beta_2) \leqslant \varepsilon$. If the ending point of α_1 (resp., β_1), is the same as the starting point of α_2 (resp., β_2), then $\delta_F(\alpha_1+\alpha_2, \beta_1+\beta_2) \leqslant \varepsilon$, where $+$ denotes the concatenation of two curves.*

3 The Decision Algorithm

Let P be a polygonal curve composed of n line segments $P_1, \ldots, P_n$, and let S be a set of k points in $\mathbb{R}^d$. In this section, we provide an algorithm to decide whether there exists a polygonal curve Q whose vertices are chosen from S, such that $\delta_F(P,Q) \leqslant \varepsilon$, for a given $\varepsilon \geqslant 0$.

We denote by s and t the starting and the ending point of P, respectively. For each segment P_i of P, we denote by C_i the cylinder $\mathcal{C}(P_i,\varepsilon)$, and by S_i the set $S \cap C_i$. Furthermore, for each point $u \in C_i$, we denote by $P_i[u]$ the line segment $P_i \cap \mathcal{B}(u,\varepsilon)$.

We call a polygonal curve Q *feasible* if all its vertices are from S, and $\delta_F(Q,P') \leqslant \varepsilon$ for a subcurve $P' \subseteq P$ starting at s. If Q ends at a point $v \in S$ and P' ends at a point $p \in P$, we call the pair (v,p) a *feasible pair*. A point $v \in S_i$ is called *reachable* (at cylinder C_i) if there is a feasible curve ending at v in C_i.

Consider a feasible curve Q starting at a point $u \in S_1$ and ending at a point $v \in S_i$. Since no backtracking is allowed in the definition of Fréchet distance, Q traverses all cylinders C_1 to C_i in order, until it reaches v. Moreover, by our definition of reachability, each vertex of Q is reachable at some cylinder C_j, $1 \leqslant j \leqslant i$.

Our approach for solving the decision problem is to process the cylinders one by one from C_1 to C_n, and identify at each cylinder C_i all points of S which are reachable at C_i. The decision problem will be then reduced (by Observation 2) to checking whether there is a reachable point in the ball $\mathcal{B}(t,\varepsilon)$.

To extend the reachability information through the cylinders, we need a primitive operation described below. Let $u \in S_i$ be a point reachable at cylinder C_i, and let Q be a feasible curve ending at u. For each point $v \in S$, we denote by $r_i(u,v)$ the index of the furthest cylinder we can reach by the curve $Q + \overrightarrow{uv}$. In other words, $r_i(u,v)$ is the largest index $h \geqslant i$ such that $v \in S_h$ is reachable via $u \in S_i$. If $Q + \overrightarrow{uv}$ is not feasible, we set $r_i(u,v) = 0$. The following lemma states how efficiently $r_i(u,v)$ can be computed. It is indeed a direct corollary of a similar lemma proved in [1] (Lemma 3) for computing the so-called right pointers.

Lemma 1 ([1]) *Given two points $u, v \in S$, we can compute $r_i(u,v)$ for all $1 \leqslant i \leqslant n$ in $O(n)$ total time.*

The following lemma will be used in our algorithm.

Lemma 2 *Let $r_i(u,v) = h$. For all $i \leqslant j \leqslant h$, if $v \in S_j$, then v is reachable at C_j.*

Proof. Let Q be a feasible curve starting at a point $w \in S \cap \mathcal{B}(s,\varepsilon)$ and ending at u, and let $Q' = Q + \overrightarrow{uv}$. Since v is reachable at C_h via Q', there is a subcurve P' of P starting at s and ending at a point $p \in P_h[v]$ (see Figure 3). Consider a moving object O walking along P from s to p, while keeping ε distance to another object O' moving monotonically along Q'. Let $q \in P_i[u]$ be the position of O when O' is at u. Fix a cylinder C_j, $i < j \leqslant h$, such that $v \in C_j$. When O reaches the point $a = \text{left}(P_j[v])$, O' is at a point $x \in \overline{uv}$ such that $\|ax\| \leqslant \varepsilon$. The subcurve of Q' from w to x has Fréchet distance at most ε to the subcurve of P from s to a, and the segment $\overline{xv}$ has Fréchet distance at most ε to the point a by Observation 1. Therefore, by Observation 2, the whole curve Q' has Fréchet distance at most ε to the subcurve P' from s to a, meaning that v is reachable at C_j. □

Note that the above proof, not only shows that v is reachable at C_j, but it also shows that the pair $(v, \text{left}(P_j[v]))$ is feasible. The following lemma is therefore immediate.

Lemma 3 *If $r_i(u,v) = h$ and $v \in S_j$, for $i < j \leqslant h$, then $(v, \text{left}(P_j[v]))$ is a feasible pair.*

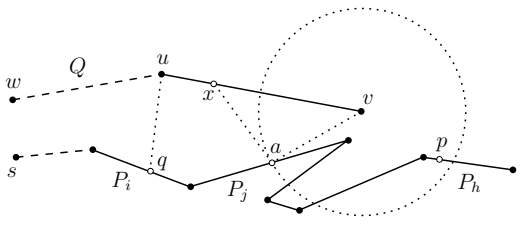

Figure 3: Proof of Lemma 2

The Algorithm Our algorithm for solving the decision problem is provided in Algorithm 1. It maintains, for each cylinder C_i, a set $\mathcal{R}_i$ of all points in S_i which are reachable at C_i. To handle the base case more easily, we assume, w.l.o.g., that the curve P starts with a segment P_0 consisting of a single point $\{s\}$. Every point of S inside the cylinder $C_0 = \mathcal{B}(s, \varepsilon)$ is reachable by definition. Therefore, we initially set $\mathcal{R}_0 = S \cap \mathcal{B}(s, \varepsilon)$ (in line 4).

For each point $v \in S$, the algorithm maintains an index ℓ_v, whose value at the beginning of each iteration i is the following: $\ell_v = \max_{0 \leqslant j < i, u \in \mathcal{R}_j} r_j(u, v)$. In other words, ℓ_v points to the largest index h for which v is reachable at C_h via a reachable point u in some earlier cylinder C_j, $j < i$. Initially, we set $\ell_v = 1$ for all points in $\mathcal{R}_0$, since all points in $\mathcal{R}_0$ are also reachable in C_1, as $C_0 \subseteq C_1$. For all other points, ℓ_v is set to 0 in the initialization step. The following invariant holds during the execution of the algorithm.

Lemma 4 *After the i-th iteration of Algorithm 1, the set $\mathcal{R}_i$ consists of all points in S_i which are reachable at cylinder C_i.*

Proof. We prove the lemma by induction on i. The base case $i = 0$ trivially holds. Suppose by induction that, for each $0 \leqslant j < i$, the set $\mathcal{R}_i$ is computed correctly. In the i-th iteration, we first add to R_i (in line 7) all points in S_i which are reachable through a point in a set $\mathcal{R}_j$, for $1 \leqslant j < i$. We call these points *entry points* of cylinder C_i. We then add to $\mathcal{R}_i$ in lines 8–11 all points in S_i which are reachable through the entry points of C_i (see Figure 4 for an example).

We first show that all points added to R_i are reachable at C_i. For each point $v \in S_i$ added to R_i in line 7, we have $\ell_v \geqslant i$. It means that there is a point $u \in R_j$, for some $j < i$, such that $r_j(u, v) \geqslant i$. Therefore, Lemma 2 implies that v is reachable at C_i. Now, consider a point v added to $\mathcal{R}_i$ in line 11. According to the condition in line 10, there is an entry point w in C_i such that $\text{left}(P_i[w]) \preceq \text{right}(P_i[v])$. By Observation 1, the segment $\overrightarrow{wv}$ is within ε Fréchet distance to the line segment from $\text{left}(P_i[w])$ to $\text{right}(P_i[v])$. Moreover, by Lemma 3, $(w, \text{left}(P_i[w]))$ is a feasible pair. Therefore, by Observation 2, v is reachable.

Next, we show that any reachable point at C_i is added to $\mathcal{R}_i$ by the algorithm. Suppose that there is a point $v \in S_i$ which is reachable at C_i, but is not added to $\mathcal{R}_i$. Let Q be a feasible curve ending at v, and w be the first point on Q which is reachable at C_i. By our definition, w is an entry point of C_i. If $w = v$, then v must be added to R_i in line 7, which is a contradiction. If w is before v on Q, then we have $\text{left}(P_i[w]) \preceq \text{right}(P_i[v])$. Now, by our selection of q in line 8, we have $q \preceq \text{left}(P_i[w]) \preceq \text{right}(P_i[v])$, and hence, v is added to R_i in line 11, which is again a contradiction. □

Algorithm 1 DECISION(S, P, ε)

1: **Initialize:**
2: compute $r_i(u, v)$ for all $u, v \in S$ and $1 \leqslant i \leqslant n$
3: set $\ell_v = 0$ for all $v \in S$
4: let $\mathcal{R}_0 = S \cap \mathcal{B}(s, \varepsilon)$
5: set $\ell_v = 1$ for all $v \in \mathcal{R}_0$
6: **for** $i = 1$ to n **do**
7: let $\mathcal{R}_i = \{v \in S_i : \ell_v \geqslant i\}$
8: let $q = \min_{v \in \mathcal{R}_i} \text{left}(P_i[v])$
9: **for all** $v \in S_i \setminus \mathcal{R}_i$ **do**
10: **if** $q \preceq \text{right}(P_i[v])$ **then**
11: add v to $\mathcal{R}_i$
12: **for all** $(u, v) \in \mathcal{R}_i \times S$ **do**
13: $\ell_v \leftarrow \max\{\ell_v, r_i(u, v)\}$
14: **return** YES if $\mathcal{R}_n \cap \mathcal{B}(t, \varepsilon) \neq \emptyset$

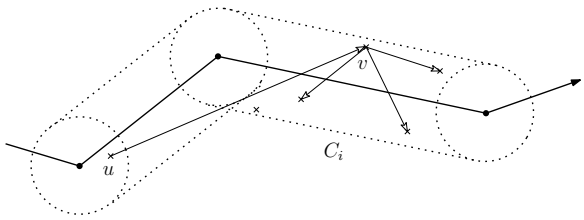

Figure 4: Point v is an entry point of C_i.

Theorem 5 *Given a polygonal curve P of n segments and a set S of k points in $\mathbb{R}^d$, we can decide in $O(nk^2)$ time whether there is a polygonal curve Q through S such that $\delta_F(P, Q) \leqslant \varepsilon$, for a given $\varepsilon \geqslant 0$. A polygonal curve Q through S of size $O(\min\{n, k\})$ that minimizes $\delta_F(P, Q)$ can be computed in $O(nk^2 \log(nk))$ time.*

Proof. The correctness of the decision algorithm (Algorithm 1) directly follows from Lemma 4. Line 2 of the algorithm takes $O(nk^2)$ time by Lemma 1. The other three lines in the initialization step (lines 3–5) take only $O(k)$ time. In the main loop, lines 7–11 take $O(k)$ time, and lines 12–13 require $O(k^2)$ time. Therefore, the whole loop takes $O(nk^2)$ time in total.

Once the algorithm finds a reachable point $v \in S_n \cap \mathcal{B}(t, \varepsilon)$, we can construct a feasible curve Q ending at v by keeping, for each reachable point u at a cylinder C_i, a back pointer to a reachable point w at C_j, $j \leqslant i$, from which u is reachable. The feasible curve Q can be then constructed by following the back pointers from v to a point in $S_1 \cap \mathcal{B}(s, \varepsilon)$. Since at most two points from each cylinder are selected in this process, the curve Q has $O(\min\{n, k\})$ segments. For the optimization problem, we use parametric search as in [1, 2], to find a curve minimizing $\delta_F(P, Q)$ by an extra $\log(nk)$-factor in $O(nk^2 \log(nk))$ time. □

4 Conclusions

In this paper, we presented a simple efficient algorithm for finding a polygonal curve through a given point set S in $\mathbb{R}^d$ such that its Fréchet distance to a given polygonal curve P is minimized. Several interesting problems remain open. For a fixed ε, one can easily modify the algorithm provided in this paper to find a curve with a minimum number of segments, having Fréchet distance at most ε to P. It can be done by keeping reachable points in a priority queue, and propagating the reachability information in a Dijkstra-like manner. However, we cannot see any easy adaption of our algorithm to find a curve passing through a maximum number of points for a fixed ε. Another major open problem is whether an efficient algorithm exists for computing a curve passing through "all" points of S with a minimum Fréchet distance to P.

The algorithm presented in this paper improves the map matching algorithm of Alt *et al.* [1] for the case of matching a curve in a complete graph. The current lower bound available for the problem is $\Omega((n + k)\log(n + k))$ due to Buchin *et al.* [3]. It is therefore open whether a better algorithm is available, or whether the algorithm obtained in this paper is optimal.

References

[1] H. Alt, A. Efrat, G. Rote, and C. Wenk. Matching planar maps. *J. Algorithms*, 49(2):262–283, 2003.

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

[3] K. Buchin, M. Buchin, C. Knauer, G. Rote, and C. Wenk. How difficult is it to walk the dog? In *Proc. 23rd European Workshop Comput. Geom.*, pages 170–173, 2007.