Fast Fréchet Distance between Curves with Long Edges

Joachim Gudmundsson
University of Sydney
Sydney, Australia
joachim.gundmundsson@sydney.edu.au

Majid Mirzanezhad
Tulane University
New Orleans, Louisiana, U.S.
mmirzane@tulane.edu

Ali Mohades
Amirkabir University of Technology
Tehran, Iran
mohades@aut.ac.ir

Carola Wenk
Tulane University
New Orleans, Louisiana, U.S.
cwenk@tulane.edu

ABSTRACT

Computing Fréchet distance between two curves takes roughly quadratic time. The only strongly subquadratic time algorithm has been proposed in [7] for c-packed curves. In this paper, we show that for curves with long edges the Fréchet distance computations become easier. Let P and Q be two polygonal curves in $\mathbb{R}^d$ with n and m vertices, respectively. We prove three main results for the case when all edges of both curves are long compared to the Fréchet distance between them: (1) a linear-time algorithm for deciding the Fréchet distance between two curves, (2) an algorithm that computes the Fréchet distance in $O((n+m)\log(n+m))$ time, and (3) a linear-time $\sqrt{d}$ -approximation algorithm for approximating the Fréchet distance between two curves.

CCS CONCEPTS

• **Theory of computation → Randomness, geometry and discrete structures**; *Computational Geometry*;

KEYWORDS

Computational Geometry, polygonal curve, Fréchet distance , approximation algorithm

ACM Reference Format:
Joachim Gudmundsson, Majid Mirzanezhad, Ali Mohades, and Carola Wenk. 2018. Fast Fréchet Distance between Curves with Long Edges. In *IWISC 2018: 3rd International Workshop on Interactive and Spatial Computing, April 12–13, 2018, Richardson, TX, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3191801.3191811

1 INTRODUCTION

Inspired by understanding behavioral ecology of groups of migration flyways derived from seagull movement data, we study the problem of similarity between trajectories with long edges. In a particular application, one might be interested in detecting groups of different migration patterns over a long-distance flying birds from some part of a country to another part.

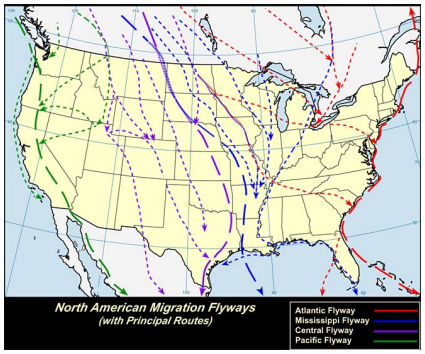


Figure 1: There are four typical flyways for seagulls to fly across the U.S. Computing the distance between different movements data results in classifying them into the suitable flyway [1].

Different flyways are relatively straight, see Fig. 1, and the trajectory data of individual birds usually consists of only one GPS sample per day in order to conserve battery power. Infrequent sampling and the straight flyways therefore result in curves with long edges, and it is desirable to compare the routes of different animals. This typical behavior of movements leads us to consider the problem of computing the distance between similar trajectories having relatively long edges. Measuring the similarity between two polygonal curves is an important problem that has applications in many areas, e.g., in morphing [8], movement analysis [9], handwriting recognition [11] and protein structure alignment [10]. One of the most popular similarity measures is the Fréchet distance, which has received considerable attention in recent years.

1.1 Background

Alt and Godau [3] were the first to describe a $O(n^2 \log n)$-time algorithm to compute the Fréchet distance between two polygonal curves with total complexity n. A first lower bound of $\Omega(n \log n)$ was given by Buchin et al. [5]. Then Bringmann [4] showed that, assuming the Strong Exponential Time Hypothesis, the Fréchet

IWISC 2018, April 12–13, 2018, Richardson, TX, USA

ACM ISBN 978-1-4503-5439-4/18/04…$15.00
https://doi.org/10.1145/3191801.3191811

distance cannot be computed in strongly subquadratic time, i.e., in time $O(n^{2-\phi})$ for any $\phi > 0$. For the *discrete* Fréchet distance Agarwal et al. [2] achieved a subquadratic running time of $O(n^2 \frac{\log\log n}{\log n})$. This result was later extended by Buchin et al. [6] to the continuous setting where they showed that the decision version can be solved in $O(n^2(\log\log n)^{3/2}/\sqrt{\log n})$ expected time. Using this, they computed the exact Fréchet distance in $O(n^2\sqrt{\log n}(\log\log n)^{3/2})$ expected time. For some special cases, such as c-packed curves, a $(1+\varepsilon)$-approximation can be computed in $O(cn/\varepsilon + cn\log n)$ time [7].

1.2 Our Results

Let P and Q be two polygonal curves in $\mathbb{R}^d$ and $\Delta > 0$. In this paper we consider the decision, optimization and approximation problems for the Fréchet distance between P and Q, all for the case where all edges of both curves are relatively long. In Section 3 we present a greedy linear-time algorithm for deciding whether the Fréchet distance is at most Δ, as long as all edges in P are at least 2Δ long and edges in Q are at least $(1+\sqrt{d})\Delta$ long. We also give an algorithm for computing the Fréchet distance in $O((n+m)\log(n+m))$ time. In Section 4 we give a linear-time algorithm to approximate the Fréchet distance up to a factor of $\sqrt{d}$.

Our results provide a first formal treatment confirming the common intuition that for very small values of Δ (relatively smaller than the shortest edge length of either curves) it should be possible to solve the Fréchet decision problem significantly faster than quadratic time.

2 PRELIMINARIES

Let $P = \langle p_1, \ldots, p_n\rangle$ be a polygonal curve with n vertices. We treat a curve as a continuous map $P : [1, n] \to \mathbb{R}^d$. In this map, $P(i) = p_i$ for an integer i, and the i-th edge is linearly parametrized as $P(i+\lambda) = (1-\lambda)p_i + \lambda p_{i+1}$, for integer i and $0 < \lambda < 1$. A re-parametrization $\sigma : [0,1] \to [1,n]$ of a curve P is any continuous, non-decreasing function such that $\sigma(0) = 1$ and $\sigma(1) = n$. $P[\sigma(a), \sigma(b)]$ denotes the subcurve of P in between $P(\sigma(a))$ and $P(\sigma(b))$ for any $0 \le a < b \le 1$. Now consider two polygonal curves $P = \langle p_1, p_2, \cdots, p_n\rangle$ and $Q = \langle q_1, q_2, \cdots, q_m\rangle$. Let l_P denote the length of the shortest edge in P, and l_Q the length of the shortest edge in Q. For two points $x, y \in \mathbb{R}^d$ and a line segment $Q \subseteq \mathbb{R}^d$, let $\|x, y\|$ denote the Euclidean distance between the points, and let $\|x, Q\| = \min_{q\in Q}\|x, q\|$ denote the minimum distance from x to any point on Q. A monotone matching between P and Q is a pair of re-parameterizations (σ, θ). The Fréchet distance between two curves is defined as $\delta_F(P,Q) = \inf_{(\sigma,\theta)} \max_t \|P(\sigma(t)), Q(\theta(t))\|$, where (σ, θ) is a monotone matching. Throughout the paper we use the following notations: Let $B(p, \Delta) = \{x \in \mathbb{R}^d \mid \|p, x\| \le \Delta\}$ be the ball with radius Δ that is centered at a point p.

Consider an edge $Q = Q[1, 2]$. P is Q-monotone if for each edge $P[i, i+1]$ the smallest angle described by vectors $P[i, i+1]$ and $Q[1, 2]$ does not exceed $\pi/2$. We call the set of points in $\mathbb{R}^d$ within distance Δ from $Q = Q[1, 2]$ the cylinder $C(Q, \Delta)$. Let q_1' be the parameter where $Q(q_1')$ is the intersection of Q with the boundary of $B(q_1, \Delta)$. Similarly define q_2' w.r.t. Q and $B(q_2, \Delta)$. α is a parameter where $P(\alpha)$ is the first point along P that meets $B(q_2, \Delta)$. Let $\mathcal{H}_1$ and

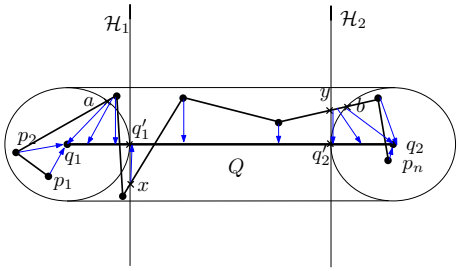


Figure 2: The blue arrows indicate orthogonal matching between P and Q.

$\mathcal{H}_2$ be a pair of hyperplanes that are perpendicular to Q and pass through q_1' and q_2', respectively. Let x be the largest parameter such that $P(x)$ intersects $\mathcal{H}_1$ and, let y be the smallest parameter such that $P(y)$ intersects $\mathcal{H}_2$. Similarly, let a be the smallest parameter such that $P(a)$ exits $B(q_1, \Delta)$ and let b be the largest parameter such that $P(b)$ enters $B(q_2, \Delta)$, see Fig. 2.

3 A GREEDY OPTIMIZATION ALGORITHM

In this section, we give a linear time algorithm for deciding whether the Fréchet distance between two polygonal curves P and Q in $\mathbb{R}^d$ with relatively long edges is at most Δ. To this end, we first prove several structural properties for the case that each edge in P and Q is longer than 2Δ and $(1+\sqrt{d})\Delta$, respectively in Section 3.1. Next, by exploiting the decision algorithm we give an exact algorithm computing Fréchet distance between P and Q in $O((n+m)\log(n+m))$ time .

3.1 Structural Properties

In this section we first define two complementary notions of *Longest Prefix* and *Orthogonal Matching*, which are the basis of our greedy algorithm. These notions provide us some properties that will allow us to develop a greedy algorithm that constructs a valid reparameterization by repeatedly computing a maximally reachable prefix on one of the curves which is provided in Section 3.2.

Definition 3.1 (Longest Δ-Prefix). Let $P = P[1, n]$ be a polygonal curve, Q be a line segment, and $\Delta > 0$. Define $\gamma = \max\{\, t \mid 1 \le t \le n \text{ and } \delta_F(P[1, t], Q) = \Delta\}$. We call $P[1, \gamma]$ the *longest Δ-prefix* of P with respect to Q.

Definition 3.2 (Orthogonal Matching). Let $P = P[1, n]$ be a polygonal curve, $Q = Q[1, 2]$ be a line segment, $\Delta > 0$. We say that P and Q admit an *orthogonal matching* if and only if:
(1) $\delta_F(P[1, x], Q[1, q_1']) \le \Delta$ (2) $\delta_F(P[x, y], Q[q_1', q_2']) \le \Delta$ and (3) $\delta_F(P[y, n], Q[q_2', 2]) \le \Delta$; see Fig. 2.

See Fig. 4, to realize the higher level idea of the definitions above and their application which makes the free space more simpler than general case. In fact these two notions are useful as long as both curves have long edges compared to Δ. Lemma 3.4 and Lemma 3.5 will show that for any arbitrary matching in free space we can greedily find extreme matchings by computing *longest Δ-prefixes* using *pieceswise orthogonal matching* only if curves have with long edges. We start with an important lemma that demonstrates the necessity of the orthogonal matching between curve P and segment Q in order to satisfy $\delta_F(P, Q) \le \Delta$.

Lemma 3.3 (Monotonicity). *Let $Q = Q[1,2]$ be a segment and $\Delta > 0$. Let $P[1,n]$ be a polygonal curve s.t. for any edge $e \in P[2,n-1]$, $\|e\| > 2\Delta$. Then $\delta_F(P,Q) \le \Delta$ if and only if P and Q admit an* orthogonal matching.

Proof. By Definition 3.2, if an orthogonal matching exists between P and Q then obviously $\delta_F(P,Q) \le \Delta$. For the opposite side, let $\mu = (\sigma,\theta)$ be an arbitrary monotone matching realizing $\delta_F(P,Q) \le \Delta$. Let x be the last intersection point between P and $\mathcal{H}_1$ and, let y be the first intersection point between P and $\mathcal{H}_2$. Also let a be the first point along P that leaves $B(q_1,\Delta)$ and b be last points that enters $B(q_2,\Delta)$, respectively. Note that every edge on $P[2,n-1]$ is longer than 2Δ which means $P[2,n-1]$ has to be monotone w.r.t. the line supporting Q. Otherwise cosnider a violating edge $e = P[i,i+i]$ for $1 < i < n-1$ that is not monotone w.r.t. Q. Since $\|p_i,p_{i+1}\| > 2\Delta$, thus $B(p_i,\Delta) \cap B(p_{i+1},\Delta) = \emptyset$ and the interval $I_i = B(p_i,\Delta) \cap Q$ encounters before the interval $I_{i+1} = B(p_i,\Delta) \cap Q$ along Q. This violates the monotonicity of (σ,θ). Hereby, $P[2,n-1]$ is monotone w.r.t. the line supporting Q. Clearly, $\delta_F(P[1,a],q_1) \le \Delta$ and $\delta_F(P[b,n],q_2) \le \Delta$. Also $\delta_F(P[a,x],Q[1,q_1']) \le \Delta$ and $\delta_F(P[y,b],Q[q_2',2]) \le \Delta$ hold by orthogonally projecting each point onto the subsegments of Q (see Fig. 2). Combining the latter results yields $\delta_F(P[1,x],Q[1,q_1']) \le \Delta$ and $\delta_F(P[y,n],Q[q_2',2]) \le \Delta$. At the end, $\delta_F(P[x,y],Q) \le \Delta$ since $P[x,y]$ can be matched to Q orthogonally and this completes the proof. □

In fact Lemma 3.3 shows that for a curve with long edges, the Fréchet distance to a line segment is determined by geometric monotonicity between the curve and the segment. The lemma below will be required in Lemma 3.5 to use the lower bound on length of edges on Q which is $(1+\sqrt{d})\Delta$.

Lemma 3.4 ($(\sqrt{d}\Delta)$-Ball). *Let $\Delta > 0$ and let P be a polygonal curve in $\mathbb{R}^d$ with long edges such that $l_P > 2\Delta$. Let $Q = Q[1,2]$ be a line segment of length greater than $(1+\sqrt{d})\Delta$. Assume that $P[1,\gamma]$ is the longest Δ-prefix of P w.r.t. Q, and α is the first point along P on the boundary of $B(q_2,\Delta)$. Then $P[\alpha,\gamma]$ is contained in $B(q_2,\sqrt{d}\Delta)$.*

Proof. By assumption $\|q_1,q_2\| > (1+\sqrt{d})\Delta \ge 2\Delta$, hence we know that $B(q_1,\Delta) \cap B(q_2,\Delta) = \emptyset$, thus α exists. Notice that $P[\alpha,\gamma] \in C(Q,\Delta)$. As we defined $\mathcal{H}_2$, it splits $C(Q,\Delta)$ into two parts, the part that contains q_1 and the part that contains q_2. By Lemma 3.3, $P[2,p]$ is Q-monotone, where $P(p)$ is the last vertex before $P(\gamma)$. Thus $P[\alpha,\gamma]$ must lie on the q_2 side of $\mathcal{H}_2$. Therefore the maximum possible distance between q_2 and a point in $P[\alpha,\gamma]$ is $\sqrt{d}\Delta$. □

In another crucial lemma below, we show that if $\delta_F(P,Q) \le \Delta$, then two curves P and Q admit a piecewise orthogonal matching, which can be obtained by computing longest Δ-prefixes of P w.r.t. each segment of Q.

Lemma 3.5 (Piecewise Orthogonal Matching Exists). *Let $\Delta > 0$, and let P and Q be two polygonal curves in $\mathbb{R}^d$ with long edges such that $l_P > 2\Delta$ and $l_Q > (1+\sqrt{d})\Delta$. If $\delta_F(P,Q) \le \Delta$, then $P[1,\gamma]$ as the longest Δ-prefix of P w.r.t. $Q[1,2]$ exists, $\delta_F(P[1,\gamma],Q[1,2]) \le \Delta$ and $\delta_F(P[\gamma,n],Q[2,m]) \le \Delta$.*

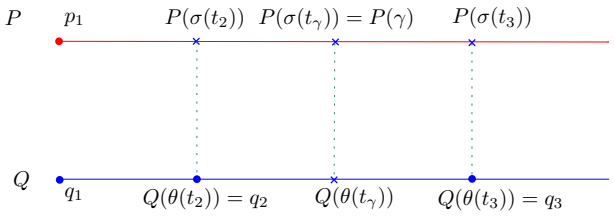


Figure 3: The Fréchet matching has to match $P(\gamma)$ to some point on $Q[2,3]$.

Proof. Let (σ,θ) be any Fréchet matching realizing $\delta_F(P,Q) \le \Delta$. There exists some $t \in [0,1]$ such that $Q(\theta(t)) = q_2$, thus $\gamma \ge \sigma(t)$ and this implies γ exists. Since $P[1,\gamma]$ is the longest Δ-prefix along P w.r.t. $Q[1,2]$ it follows by definition that $\delta_F(P[1,\gamma],Q[1,2]) \le \Delta$. In the remainder of this proof we construct a matching to prove that $\delta_F(P[\gamma,n],Q[2,m]) \le \Delta$.

Let $t_\gamma \in [0,1]$ be the largest value such that $P(\sigma(t_\gamma)) = P(\gamma)$. Let $t_2 \in [0,1]$ be the largest value such that $Q(\theta(t_2)) = q_2$. By Lemma 3.4, $P[\sigma(t_2),\gamma] \in B(q_2,\sqrt{d}\Delta)$. Now let $t_3 \in [0,1]$ be the smallest value such that $Q(\theta(t_3)) = q_3$. We have $\|q_2,q_3\| > (1+\sqrt{d})\Delta$, therefore $B(q_2,\sqrt{d}\Delta) \cap B(q_3,\Delta) = \emptyset$ and thus q_3 cannot be matched to any point in $P[\sigma(t_2),\gamma]$. Correspondingly, $\sigma(t_2) \le \gamma = \sigma(t_\gamma) < \sigma(t_3)$, and follows $\theta(t_2) \le \theta(t_\gamma) < \theta(t_3)$.

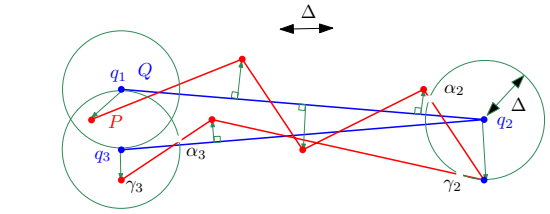
Now we construct a new Piecewise orthogonal matching $(\bar{\sigma},\bar{\theta})$ realizing $\delta_F(P[\gamma,n],Q[2,m]) \le \Delta$ as: $\bar{\sigma}(t) = \sigma(t)$ and $\bar{\theta}(t) = \theta(t)$ for all $t_\gamma \le t \le 1$. On the other hand, since $\|P(\gamma),q_2\| \le \Delta$ and $\|P(\gamma),Q(\theta(t))\| \le \Delta$, we know that $Q[2,\theta(t_\gamma)] \in B(\gamma,\Delta)$. Hence, $\bar{\sigma}(t) = \gamma$ and $\bar{\theta}(t) = \frac{t_\gamma - t}{t_\gamma}\cdot 2 + \frac{t}{t_\gamma}\cdot\theta(t_\gamma)$ for all $t_2 \le t \le t_\gamma$ (see Fig. 3). Therefore, we have $\delta_F(P[\gamma,n],Q[2,m]) \le \Delta$, which completes the proof. □

In fact, Lemma 3.5 implies that if P and Q have long edges then the free space and the reachable space are simpler than in the general case, because due the horizontal and vertical line segments of reachable space connecting to an arbitrary matching, holes are not possible (See Fig. 4).

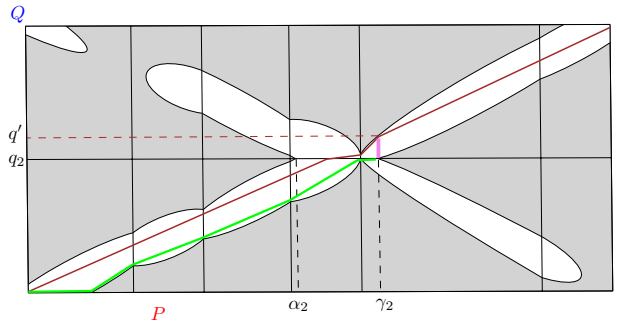
3.2 The Decider

Now we present a linear time algorithm using the properties provided in Section 3.1. The pseudocode of our Decider is presented in Algorithm 1. The input to this Decider is a set of polygonal curves P and Q, and $\Delta > 0$. The algorithm assumes that P and Q have long edges. In each iteration the function LongestDeltaPrefix returns γ for subcurve $P[\gamma_i,p_n]$, as defined in Definition 3.1, w.r.t. $Q[i,i+1]$, if it exists. If any $\gamma = null$ then "No" is returned. Otherwise, the next edge of Q is processed. This continues iteratively until all edges have been processed, or until no γ exists.

From Lemma 3.3 follows that P and Q admit an orthogonal matching if and only if two conditions holds: (1) $P[\gamma_{i-1},\gamma_i] \in C(Q[i-1,i],\Delta)$ and (2) $P[\gamma_{i-1},\gamma_i]$ is monotone w.r.t. directed line supporting $Q[i-1,i]$. Now Lemma 3.3 can be used to implement the LongestDeltaPrefix procedure as follows: Determine the first edge on $P[\gamma_{i-1},n]$ which violates the two conditions above w.r.t. $Q[i-1,i]$. If the violation occurs before reaching $B(q_i,\Delta)$, then



(a)



(b)

Figure 4: λ_2 is the longest Δ-prefix on P w.r.t. $Q[1,2]$ realizing the green (bottommost) xy-monotone envelope in the free space. (a) If $\delta_F(P,Q) \le \Delta$, then piecewise *orthogonal matching* between subpaths $(P[p_1,\gamma_2], P[\gamma_2,\gamma_3],\cdots)$ and corresponding Q's edges $(Q[1,2], Q[2,3],\cdots)$, respectively exists. (b) As shown in Lemma 3.5, finding γ_2 allows us to keep track of any arbitrary matching (brown path) by following the pink vertical line-segment connected between γ_2 and the brown matching.

Algorithm 1: Decide whether $\delta_F(P,Q) \le \Delta$

```
1 Decider(P[1,n], Q[1,m], Δ)
2     if l_P ≤ 2Δ or l_Q ≤ (1+√d)Δ then return "I don't know."
3     s ← 1
4     for i ← 2 to m do
5         γ ← LongestDeltaPrefix(P[s,n], Q[i−1,i], Δ)
6         if γ = null then return "No"
7         s ← γ
8     if γ < n then return "No"
9     return "Yes"
```

obviously $\gamma_i = null$. Otherwise we intersect the first violating edge e along $P[\gamma_{i-1}, n]$ with the boundary of $B(q_i,\Delta)$ to find γ. Clearly, the whole process takes $O(n)$ time.

Now we prove the correctness of our decision algorithm:

Theorem 3.6 (Correctness). *Let $\Delta > 0$, and let P and Q be two polygonal curves in $\mathbb{R}^d$ with long edges such that $l_P > 2\Delta$ and $l_Q > (1+\sqrt{d})\Delta$. Then Decider(P,Q,Δ) returns "Yes" if and only if $\delta_F(P,Q) \le \Delta$.*

Proof. Let γ_i be the value of γ computed in the i-th iteration of the for loop in line 6 of the algorithm. If the algorithm returns "Yes" then the sequence $\{(q_i,\gamma_i)\}$ with $i = 1,\cdots,m$ with $\gamma_1 = 1$ and $\gamma_m = n$ describes a monotone matching that realizes $\delta_F(P,Q) \le \Delta$.

If $\delta_F(P,Q) \le \Delta$, then we use Lemma 3.5 to prove by induction on i that the algorithm returns "Yes", i.e., all longest Δ-prefixes $\langle P[1,\gamma_2], P[\gamma_2,\gamma_3],\cdots, P[\gamma_{m-1},\gamma_m]\rangle$ of P w.r.t. corresponding segments of Q exist. For $i = 2$, following Lemma 3.5, γ_2 exists and can be found by the algorithm. For any $i > 2$, the algorithm has determined $\gamma_2,\cdots,\gamma_{i-1}$ already and by Lemma 3.5, $\delta_F(P[\gamma_{i-1},n], Q[i-1,m]) \le \Delta$. Another application of Lemma 3.5 yields that $\delta_F(P[\gamma_{i-1},\gamma_i], Q[i-1,i]) \le \Delta$ and $\delta_F(P[\gamma_i,n], Q[i,m]) \le \Delta$ since γ_i essentially exists.

In the case that $i = m-1$ it remains to prove that $\gamma_{i+1} = \gamma_m = n$. Assume $\gamma_m < n$. Since $P[\gamma_{m-1},\gamma_m]$ is the longest Δ-prefix, there is no other point $\gamma'_m \in (\gamma_m, n]$ such that $\delta_F(P[\gamma_{m-1},\gamma'_m], Q[m-1,m]) \le \Delta$. Consequently, $\delta_F(P[\gamma_{m-1},\gamma'_m], Q[m-1,m]) > \Delta$ and follows $\delta_F(P[\gamma_{m-1},n], Q[m-1,m]) > \Delta$. By applying modus tollen rule of Lemma 3.5 on $P[\gamma_{m-1},n]$ and $Q[m-1,m]$, $\delta_F(P,Q) > \Delta$ and that would be a contradiction. Therefore $\gamma_m = n$ and the algorithm returns "Yes" as claimed. □

As a result we have the following theorem:

Theorem 3.7 (Decider). *Let $\Delta > 0$, and let P and Q be two polygonal curves in $\mathbb{R}^d$ with long edges such that $l_P > 2\Delta$ and $l_Q > (1+\sqrt{d})\Delta$. Let n be the number of vertices in P, and m be the number of vertices in Q. Then there exists a greedy decider, Algorithm 1, that can determine $\delta_F(P,Q) \le \Delta$ in $O(n+m)$ time.*

Proof. Following Theorem 3.6, all $P(\gamma_i)$s over P are matched to the q_is over Q. This means the algorithm greedily finds the next γ_{i+1} w.r.t. each edge of Q and Δ. We have $m-1$ edges and the total time for greedily finding the longest Δ-prefixes takes $O(n)$, this implies Algorithm 1 runs in $O(n+m)$ time. □

3.3 The Optimization Algorithm

The main idea of our algorithm is that we compute critical values of the Fréchet distance between two curves and we show that the number of these values is linear. We then perform binary search on these critical values to find the optimal value acquired by the decision algorithm. The following observation allows us to compute critical values efficiently:

Observation 3.8 (Superpath). *Let $\Delta > 0$ and $\Delta^* > 0$ be two real numbers. In addition, let $\gamma_i =$ LongestDeltaPrefix$(P[\gamma_{i-1},n], Q[i-1,i],\Delta)$ and let α_i be the smallest parameter where $P(\alpha_i) \in P[\gamma_{i-1},n]$ meets $B(q_i,\Delta)$. Similarly, define γ^*_i and α^*_i w.r.t. Δ^*. If $\Delta^* \le \Delta$, then $\alpha_i \le \alpha^*_i \le \gamma^*_i \le \gamma_i$.*

In fact, Observation 3.8 concludes that the path $P[\alpha_{i-1},\gamma_i]$ is the longest superpath for $P[\alpha^*_{i-1},\gamma^*_i]$ when $\Delta = \min(l_P/2, l_Q/(1+\sqrt{d}))$ and $\Delta^* = \delta_F(P,Q)$. Now, given a segment $Q[i-1,i]$ and Δ, our optimization algorithm simply works as follows: (1) It runs Algorithm 1 with $\Delta = \min(l_P/2, l_Q/(1+\sqrt{d}))$, and only proceeds if Decider(P,Q,Δ) returns 'Yes'.

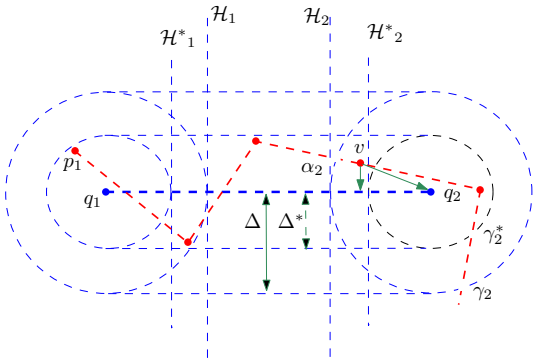


Figure 5: $\delta_F(P,Q) = \Delta^* \le \Delta$ and as cylinder $C(Q,\Delta^*)$ is unknown we are not sure whether v lies inside $B(q_2,\Delta^*)$ or not. We are aware of two possibilities for v's *point to point* distance, that is either the orthogonal one onto Q or the distance to q_2. Vertices between $\mathcal{H}_1$ and $\mathcal{H}_2$ must have orthogonal distances.

This gives us all $\gamma_i = \text{LongestDeltaPrefix}(P[\gamma_{i-1},n], Q[i-1,i], \Delta)$. (2) For every $v \in P[\alpha_{i-1},\gamma_i]$, if v lies between $\mathcal{H}_{i-1},\mathcal{H}_i$ it stores the (orthogonal) distance $\|v,Q[i-1,i]\|$ in the set C_i. Otherwise it stores the two distances $\|v,Q[i-1,i]\|$ and $(\|v,q_{i-1}\|$ or $\|v,q_i\|)$ in C_i. In the end we compute $C := \cup_{i=2}^m C_i$. The call to Algorithm 1 ensures that P and Q satisfy the constraint $l_P > 2\Delta$ and $l_Q > (1+\sqrt{d})\Delta$. Consequently, $l_P > 2\Delta_x$ and $l_Q > (1+\sqrt{d})\Delta_x$ for any critical value $\Delta_x \le \Delta$. Notice that since all γ_i exist, we know that $\gamma_i^* = \text{LongestDeltaPrefix}(P[\gamma_{i-1}^*,n], Q[i-1,i], \Delta^*)$ exist as well for all $i = 1,\cdots,m$. Nevertheless we do not know γ_i^* for all $i = 1,\cdots,m$ beforehand. But according to Observation 3.8, we know that whatever value Δ^* would be, the widest range we need to look for critical values between $P[\gamma_{i-1}^*,\gamma_i^*]$ and $Q[i-1,i]$ is $P[\alpha_{i-1},\gamma_i]$.

An orthogonal matching (Definition 3.2) matches points in such a way that there are two types of *point to point* distances: Some of these distances are obtained by a simple projection of points from P to Q, and some match points from P to the endpoints of $Q[i-1,i]$. These two types of distances are sufficient to acquire all possibilities for Δ^* since the matching changes when $C(Q,\Delta)$ shrinks to $C(Q,\Delta^*)$. (See Fig. 5 for more illustration). Note that there is no need for the values greater than Δ since $\Delta^* \le \Delta$. Once we have computed C we perform binary search to find the optimal value.

For each edge of Q we find these distances between $P[\alpha_{i-1},\gamma_i]$ and $Q[i-1,i]$ until we meet q_m. Once we have all m many sets of critical values, sorting them allows us to perform a binary search on them to find the optimal value Δ^* in which $\text{Decider}(P,Q,\Delta^*)$ returns 'Yes' but 'No' for the values below Δ^*. We can summarize this section with the following theorem:

Theorem 3.9 (Optimization). *Let P and Q be two polygonal curves in $\mathbb{R}^d$ with n and m vertices, respectively.*
If $\delta_F(P,Q) < \min\{l_P/(2), l_Q/(1+\sqrt{d})\}$, then $\delta_F(P,Q)$ can be computed in $O((n+m)\log(n+m))$ time.

Proof. Let $P[\gamma_{i-1},\gamma_i]$ have n_i vertices and assume we know $\sum_{i=2}^m n_i = O(n+m)$. When processing $Q[i,i+1]$, $P[\alpha_i,\gamma_i]$ is being revisited again. Hence, $|C_i| \le 2n_i$ (see Fig. 5). By assumption above, $C_{i+1} \le 2n_i + n_{i+1}$, therefore $C = \cup_{i=2}^m C_i \le \sum_{i=2}^m (4n_i + n_{i+1}) < 5(n+m) = O(n+m)$. Sorting C and performing binary search on it takes $O((n+m)\log(n+m))$ time. □

4 AN APPROXIMATION ALGORITHM

In this section, we present an algorithm approximating the Fréchet distance between two curves with long edges, respectively. We present $\sqrt{d}$-approximation algorithm running in linear time. Similar to Section 3, we introduce a variant notion of Definition 3.1 called *minimum prefix*:

Definition 4.1 (Minimum Prefix). Let P be a polygonal curve and Q be a segment. Define $\gamma' = \max\{\, t \mid \text{argmin}_{1\le t\le n} \delta_F(P[1,t],Q)\}$. We call $P[1,\gamma']$ the *minimum prefix of P with respect to Q*.

The approximation algorithm is presented in Algorithm 2. First, for an initial threshold Δ_0 less than but sufficiently close to $\min\{l_P/(2\sqrt{d}), l_Q/(2d)\}$, it runs Decider (P,Q,Δ_0). It only continues if "Yes" gets returned. This ensures that P and Q have long edges, with $l_P > 2\sqrt{d}\Delta > 2\Delta$ and $l_Q > 2d\Delta > (1+\sqrt{d})\Delta$. Then, similar to the decision algorithm, the approximation algorithm greedily searches for longest Δ-prefixes for each segment of Q. However, it updates the current value of Δ in each step, by computing the minimum prefix and its associated Fréchet distance to the portion of Q considered so far. For a line segment Q, the function MinimumPrefix$(P[1,n],Q)$ returns (γ,Δ) such that $P[1,\gamma]$ is the minimum prefix of $P[1,n]$ with respect to Q, and $\Delta = \delta_F(P[1,\gamma],Q)$.

Algorithm 2: Compute $\delta_F(P,Q)$

```
1  ApproximationAlgorithm(P[1,n], Q[1,m])
2      Δ0 ← min(lP/2√d, lQ/2d)
3      if Decider(P, Q, Δ0) = "No" then return "I don't know."
4      (γ2, Δ2) ← MinimumPrefix(P[1,n], Q[1,2])
5      Δ ← Δ2
6      s ← γ2
7      for i ← 3 to m do
8          (γi, Δi) ← MinimumPrefix(P[s,n], Q[i−1,i])
9          Δ ← max(Δ, Δi)
10         s ← γi
11     if γm = n then
12         return Δ
13     else
14         Δ ← max(Δ, δF(P[γm,n], qm))
15         return Δ
```

When Δ is increased, the longest Δ-prefix can only get longer, as summarized below.

Observation 4.2. *Let $P = P[1,n]$ and $Q = Q[1,m]$ be two polygonal curves and let $\Delta > 0$.*
For any $x, y \in P$, let $\gamma = \text{LongestDeltaPrefix}(P[x,n], Q[i-1,i], \Delta))$ and $\gamma' = \text{LongestDeltaPrefix}(P[y,n], Q[i-1,i], \Delta')$. Then $\Delta < \Delta'$ if and only if $\gamma < \gamma'$.

Now we are ready to prove the correctness of Algorithm 2:

Lemma 4.3 (The Approximation). *Let $P = P[1,n]$ and $Q = Q[1,m]$ be two polygonal curves and let $\Delta^* = \delta_F(P,Q)$. If $\Delta^* < \min\{l_P/(2\sqrt{d}), l_Q/(2d)\}$ then ApproximationAlgorithm(P,Q) returns $\sqrt{d}\Delta^*$. Otherwise it returns "I don't know.".*

Proof. From the algorithm: $\Delta_i = \delta_F(P[\gamma_{i-1},\gamma_i], Q[i-1,i])$. Now suppose $\delta_F(P,Q) = \Delta^*$. We prove by induction on i that $\Delta_i \le \sqrt{d}\Delta^*$. For $i = 2$, Δ_2 is being minimized and obviously $\Delta_2 \le \Delta^* < \sqrt{d}\Delta^*$. For any $i > 2$, there are two possible cases: If $\Delta_i \le \Delta^*$, then trivially $\Delta_i < \sqrt{d}\Delta^*$. In the remainder of the proof we consider the case that $\Delta_i > \Delta^*$. We know from the proof of Theorem 3.6 that all $\gamma_i^* = \text{LongestDeltaPrefix}(P[\gamma_{i-1}^*, n], Q[i-1,i], \Delta^*)$ for all $i = 1, 2, \cdots m$ exist. And by inductive hypothesis we know that $\max(\Delta_2, \cdots, \Delta_{i-1}) \le \sqrt{d}\Delta^*$.

In order to show that $\Delta_i \le \sqrt{d}\Delta^*$, consider an optimal matching (σ,θ) realizing Δ^*. For the sake of contradiction we assume $\Delta_i > \sqrt{d}\Delta^*$. By Observation 4.2, it follows from $\Delta^* < \Delta_i$ that $\gamma_i^* < \gamma_i$. We now distinguish two subcases. (2a) If $\gamma_{i-1} < \gamma_{i-1}^*$, then by Lemma 3.4 we have $\delta_F(P[\gamma_{i-1},\gamma_{i-1}^*], q_{i-1}) \le \sqrt{d}\Delta^*$. Also $\delta_F(P[\gamma_{i-1}^*,\gamma_i^*], Q[i-1,i]) \le \Delta^*$, hence $\delta_F(P[\gamma_{i-1},\gamma_i^*], Q[i-1,i]) \le \sqrt{d}\Delta^*$ and $(\Delta_i,\gamma_i) = (\sqrt{d}\Delta^*, \gamma_i^*)$ would be returned by MinimumPrefix, which contradicts $\Delta_i > \sqrt{d}\Delta^*$ (see Fig. 6.a). (2b) Now for the case that $\gamma_{i-1}^* < \gamma_{i-1}$, consider the same matching as above realizing $\delta_F(P,Q) = \Delta^*$. There exists a $t \in [0,1]$ such that $\gamma_{i-1} = \sigma(t)$. Clearly $Q(\theta(t)) \in Q[i-1,i]$ because $B(q_{i-1},\Delta_{i-1}) \cap B(q_i,\Delta^*) = \emptyset$ since $\|q_{i-1},q_i\| > 2d$ and $\Delta_{i-1} \le \sqrt{d}\Delta^*$. This implies $\gamma_{i-1} < \gamma_i^*$. Hence, $\gamma_{i-1}^* < \gamma_{i-1} < \gamma_i^*$ and correspondingly $q_{i-1} \le Q(\theta(t)) \le q_i$. By induction we know $\Delta_{i-1} = \|q_{i-1}, P(\gamma_{i-1})\| \le \sqrt{d}\Delta^*$, thus $Q[i-1,\theta(t)] \subseteq B(P(\gamma_{i-1}), \sqrt{d}\Delta^*)$ which implies $\delta_F(P(\gamma_{i-1}), Q[i-1,\theta(t)]) \le \sqrt{d}\Delta^*$.

Combining the latter inequality with $\delta_F(P[\gamma_{i-1},\gamma_i^*], Q[\theta(t),i]) \le \Delta^*$ from the optimal matching, implies that $(\Delta_i,\gamma_i) = (\sqrt{d}\Delta^*,\gamma_i^*)$ must be returned by MinimumPrefix, which is again a contradiction. At the end, if $\gamma_m < n = \gamma_m^*$, then Lemma 3.4 again implies $\delta_F(P[\gamma_m,n], q_m) \le \sqrt{d}\Delta^*$ as claimed (see Fig. 6.b). □

The MinimumPrefix Procedure: We implement MinimumPrefix$(P[1,n],Q)$ for a line segment $Q = Q[1,2]$ as follows: For every $i \in \{1,\ldots,n-1\}$ let c_i be the distance associated with a minimum prefix ending on the segment $P[i,i+1]$. Formally, $c_i = \min_{t\in[i,i+1]} \delta_F(P[1,t],Q)$. Algorithm 3 computes all the c_i in a dynamic programming fashion. The minimum of the c_i is the desired Δ, and the LongestDeltaPrefix computes the corresponding γ. The following lemma shows the correctness of Algorithm 3:

Lemma 4.4 (Correctness). *Let $Q = Q[1,2]$ be a line segment and let $P = P[1,n']$ be a Q-monotone curve.*

If $\delta_F(P,Q) < \min\{l_P/(2\sqrt{d}), l_Q/(2d)\}$, then the distance returned by MinimumPrefix(P,Q) is $\min_{1\le t\le n'} \delta_F(P[1,t],Q)$.

Proof. According to the algorithm,

$$c_i = \max\{\|p_1,q_1\|, \max_{j=1}^{i-1} \|p_{j+1},Q\|, \|P[i,i+1],q_2\|\}.$$

Since Q is a segment and P is Q-monotone, the following follows for any $i \le t \le i+1$ using simple projections of points from P onto Q:

$$\delta_F(P[1,t],Q) = \max\{\|p_1,q_1\|, \max_{j=1}^{i-1} \|p_{j+1},Q\|, \|P(t),q_2\|\}$$

By taking the minimum on both sides: $\min_{i\le t\le i+1} \delta_F(P[1,t],Q) = \max\{\|p_1,q_1\|, \max_{j=1}^{i-1} \|p_{j+1},Q\|, \min_{i\le t\le i+1} \|P(t),q_2\|\} = c_i$.

It suffices to run the for-loop until $n' = \lfloor\gamma'\rfloor$, since by assumption we only compute the minimum Δ-prefix $P[1,\gamma]$ if its distance is at most Δ', and from Observation 4.2 follows $\gamma < \gamma'$. Therefore, $\Delta = \min_{i=1}^{\lfloor\gamma'\rfloor} c_i = \min_{i=1}^{n'} c_i = \min_{1\le t\le n'} \delta_F(P[1,t],Q)$. □

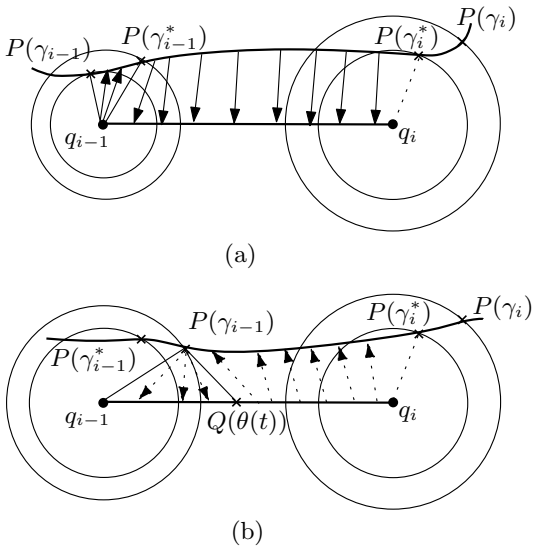


Figure 6: Illustration of the proof of Lemma 4.3 for the case $\Delta_i > \Delta^*$. (a) $\gamma_{i-1} < \gamma_{i-1}^*$ (b) $\gamma_{i-1} > \gamma_{i-1}^*$.

Algorithm 3: Compute MinimumPrefix$(P[1,n],Q[1,2])$

```
1 MinimumPrefix(P[1,n], Q[1,2])
2     c ← ||p_1, q_1||
3     Δ' ← min{l_P/2, l_Q/(2√d)}
4     γ' ← LongestDeltaPrefix(P[1,n], Q, Δ')
5     for i ← 1 to ⌊γ'⌋ do
6         c_i ← max{c, ||P[i, i+1], q_2||}
7         c ← max{c, ||p_{i+1}, Q||}
8     Δ = min_{i=1}^{⌊γ'⌋} c_i
9     return (Δ, LongestDeltaPrefix(P[1,n], Q, Δ))
```

Theorem 4.5 (Runtime). *Let P and Q be two polygonal curves in $\mathbb{R}^d$ with n and m vertices, respectively.*

If $\delta_F(P,Q) < \min\{l_P/(2\sqrt{d}), l_Q/(2d)\}$, then Algorithm 2 approximates the Fréchet distance between P and Q in $O(n+m)$ time within an approximation factor of $\sqrt{d}$.

Proof. Let $\Delta^* = \delta_F(P,Q)$. The algorithm only proceeds past line 3 if $\Delta^* \le \Delta_0 = \min\{l_P/(2\sqrt{d}), l_Q/(2d)\}$.

Now, as $\Delta' = \sqrt{d}\Delta_0$, $\gamma'_1 = 1$, and for all $j = 2, \cdots, m$, let $\gamma'_j = \textsc{LongestDeltaPrefix}(P[\gamma'_{j-1}, n], Q[j-1, j], \Delta')$. Note that by definition of Δ', both curves have long edges, i.e., $l_P > 2\Delta'$ and $l_Q > (1+\sqrt{d})\Delta'$. For each segment $Q[i-1, i]$ with $i > 1$, the MinimumPrefix procedure first computes γ'_i (see line 4 in Algorithm 3), and finally returns γ_i as the endpoint of the MinimumPrefix subcurve w.r.t. $Q[i-1, i]$. From the proof of Lemma 4.3 we know that $\Delta_i \leq \sqrt{d}\Delta^* \leq \sqrt{d}\Delta_0 = \Delta'$ for all i. Thus, Observation 4.2 implies that $\gamma_i \leq \gamma'_i$. Also note that $\gamma_i > \gamma_{i-1}$ since $P(\gamma_{i-1}) \in B(q_{i-1}, \Delta_{i-1})$ and $P(\gamma_i) \in B(q_i, \Delta_i)$, and moreover $B(q_{i-1}, \Delta_{i-1}) \cap B(q_i, \Delta_i) = \emptyset$ because $\|q_i, q_{i-1}\| > 2d\Delta_0 \geq 2d\Delta^* > 2(\sqrt{d}\Delta^*) \geq 2(\max(\Delta_i, \Delta_{i-1}))$. Hence $\gamma_{i-1} < \gamma_i \leq \gamma'_i$.

The for-loop in Algorithm 3 has n_i iterations, where n_i is the number of integers in the interval $[\gamma_{i-1}, \gamma'_i]$, and $\sum_{i=2}^{m} n_i = n$. For the segment $Q[i, i+1]$, MinimumPrefix starts with γ_i to find γ_{i+1} where $\gamma_{i+1} \leq \gamma'_{i+1}$. Overall, MinimumPrefix processes the portion $[\gamma_i, \gamma'_i]$ twice, and computing the minimum of the c_i takes linear time, thus MinimumPrefix takes at most $O(n_i)$ time. Therefore, Algorithm 2 takes $O(n + m)$ time. □

5 DISCUSSION AND FUTURE WORK

In this paper we provided a linear time decision algorithm, an $O((n + m)\log(n + m))$ time optimization algorithm, and a linear time $\sqrt{d}$-approximation algorithm between curves that have long edges. Our algorithms are simple greedy algorithms that run in any constant dimension. One interesting problem would be to develop a trade-off between the length of edges and the runtime, and in general prove hardness in terms of the edge lengths.

Acknowledgements: The authors would like to acknowledge the generous support of the Australian Research Council's Discovery Projects funding scheme (DP150101134) and the National Science Foundation under grant CCF-1637576. We particularly, thank an anonymous reviewer for insightful comments that helped us to improve the algorithm in Section 3.

REFERENCES

[1] 2016. North America Migration Flyways. (sep 2016). https://www.nps.gov/pais/learn/nature/birds.htm

[2] P.K. Agarwal, R.B. Avraham, H. Kaplan, and M. Sharir. 2014. Computing the Discrete Fréchet Distance in Subquadratic Time. *SIAM J. Comput.* 43 (2014), 429–449.

[3] H. Alt and M. Godau. 1995. Computing the Fréchet Distance between two Polygonal Curves. *International Journal of Computational Geometry and Applications* 5, 1–2 (1995), 75–91.

[4] K. Bringmann. 2014. Why Walking the Dog Takes Time: Fréchet Distance Has no Strongly Subquadtatic Algorithms unless SETH Fails. In *Proceedings of the 55th IEEE Symposium on Foundations of Computer Science (FOCS)*. 226–236.

[5] K. Buchin, M. Buchin, C. Knauer, G. Rote, and C. Wenk. 2007. How difficult Is It to Walk you Dog?. In *32nd European Workshop on Computational Geometry*. 170–173.

[6] K. Buchin, M. Buchin, W. Meulemans, and W. Mulzer. 2014. Four Soviets Walk the Dog - with an Application to Alt's Conjecture. In *Proceeding of the 25th ACM-SIAM Symposium on Discrete Algorithms*. 1399–1413.

[7] A. Driemel, S. Har-Peled, and C. Wenk. 2012. Approximating the Fréchet Distance for Realistic Curves in Near Linear Time. *Discrete & Computational Geometry* 48 (2012), 94–127.

[8] A. Efrat, L.J. Guibas, S. Har-Peled, J.S.B Mitchell, and T.M. Murali. 2002. New Similarity Meaures between Polylines with Applicatons yo Morphing and Polygon Sweeping. *Discrete & Computational Geometry* 28, 4 (2002), 535–569.

[9] J. Gudmundsson, P. Laube, and T. Wolle. 2007. *Movement Pattern in Spatio Temporal Data* (1st ed.). Springer-Verlag.

[10] M. Jiang, Y. Xu, and B. Zhu. 2008. Protein Structure–Structure Alignment with Discrete Fréchet Distance. *Journal of Bioinformatics and Computational Biology* 6, 1 (2008), 51– 64.

[11] R. Sriraghavendra, K. Karthik, and C. Bhattacharyya. 2007. Fréchet Distance based Approach for Searching Online Handwrittien Documents. In *In Proceeding of 9th International Conference on Document Analysis and Recognition*. 461–465.