# Visualization of the Fréchet distance using free space diagrams

Lazar Savić
Faculty of Mathematics, University of Belgrade

December 28, 2025

## Contents



**Abstract**

The Fréchet distance is a similarity measure between curves that takes into account their geometric structure and ordering. Unlike simpler distance measures, it captures how two curves can be traversed continuously and monotonically. In this document, we present an interactive visualization tool for the continuous Fréchet distance based on free space diagrams. The tool allows users to load polygonal curves, explore the free space structure for different distance thresholds, and observe an animation that illustrates the optimal traversal realizing the Fréchet distance.

## 1 Introduction

Comparing curves is a fundamental problem in computational geometry, with applications in trajectory analysis, handwriting recognition, map matching, and motion planning [2]. A widely used similarity measure for this purpose is the *Fréchet distance*, which intuitively measures how similar two curves are while respecting the order in which points are traversed.

The Fréchet distance is often explained using the metaphor of a man walking his dog, where each walks along a separate path without backtracking. It is defined as the minimal length of the leash required for the man and the dog to traverse two given polylines. While intuitive, the formal definition and computation of the Fréchet distance can be difficult to grasp without visual support.

This project focuses on building an interactive visualization tool that illustrates an approximation of the continuous Fréchet distance using *free space diagrams*. The tool enables users to load polygonal curves, adjust the distance threshold, and visually inspect reachability and the resulting optimal traversal. Since obtaining the exact value of the continuous Fréchet distance is challenging, the software primarily addresses the decision version of the problem: determining whether a traversal of the polygonal curves is possible for a given leash length. By interactively experimenting with different leash lengths, users can nevertheless obtain an approximate value of the distance.

# 2 Problem definition

## 2.1 Polygonal curves

A polygonal curve $P$ in the plane is defined as an ordered sequence of points

$$P = (P_0, P_1, \ldots, P_m),$$

where consecutive points are connected by straight-line segments [2]. Similarly, a second curve $Q = (Q_0, Q_1, \ldots, Q_n)$ is defined.

Each curve can be parametrized continuously over the interval $[0, m]$ or $[0, n]$ respectively, where the parameter corresponds to a position along the curve, potentially lying in the interior of a segment.

## 2.2 Fréchet distance

The continuous Fréchet distance [1] between two curves $P$ and $Q$ is defined as

$$\delta_F(P, Q) = \inf_{\alpha,\beta} \max_{t\in[0,1]} \|P(\alpha(t)) - Q(\beta(t))\|,$$

where $\alpha$ and $\beta$ are continuous, non-decreasing reparameterizations of the interval $[0, 1]$ onto the parameter domains of $P$ and $Q$.

This definition enforces that both curves are traversed monotonically and continuously, making the Fréchet distance sensitive to both geometry and ordering.

# 3 Free space diagram

## 3.1 Definition

The free space diagram provides a geometric interpretation of the Fréchet distance. It is defined over the parameter space $[0, m] \times [0, n]$, where each point $(s, t)$ corresponds to the pair of points $P(s)$ and $Q(t)$.

For a given distance threshold $\varepsilon$, the free space [1] is defined as

$$F_\varepsilon = \{(s, t) \mid \|P(s) - Q(t)\| \leq \varepsilon\}.$$

The parameter space is divided into rectangular cells, each corresponding to a pair of segments from $P$ and $Q$. Within each cell, the free space is described by a (possibly empty) elliptical region restricted to the cell. This region may intersect the cell boundaries, thereby inducing free space intervals on the cell edges. They are denoted in blue on the free space diagram (Fig. 1).

### 3.2 Reachability

The key question is whether there exists a continuous, monotone path from $(0, 0)$ to $(m, n)$ that lies entirely within the free space. Such a path corresponds to a valid traversal of both curves under distance $\varepsilon$.

Reachability is computed by propagating reachable intervals across adjacent cells in the free space diagram. If the top-right corner $(m, n)$ is reachable, then $\varepsilon$ is a valid upper bound on the Fréchet distance. The smallest value of $\varepsilon$ for which this becomes possible is the Fréchet distance.

Intervals that are reachable from the bottom left corner of the free space diagram with the given leash length are denoted in green (Fig. 1).

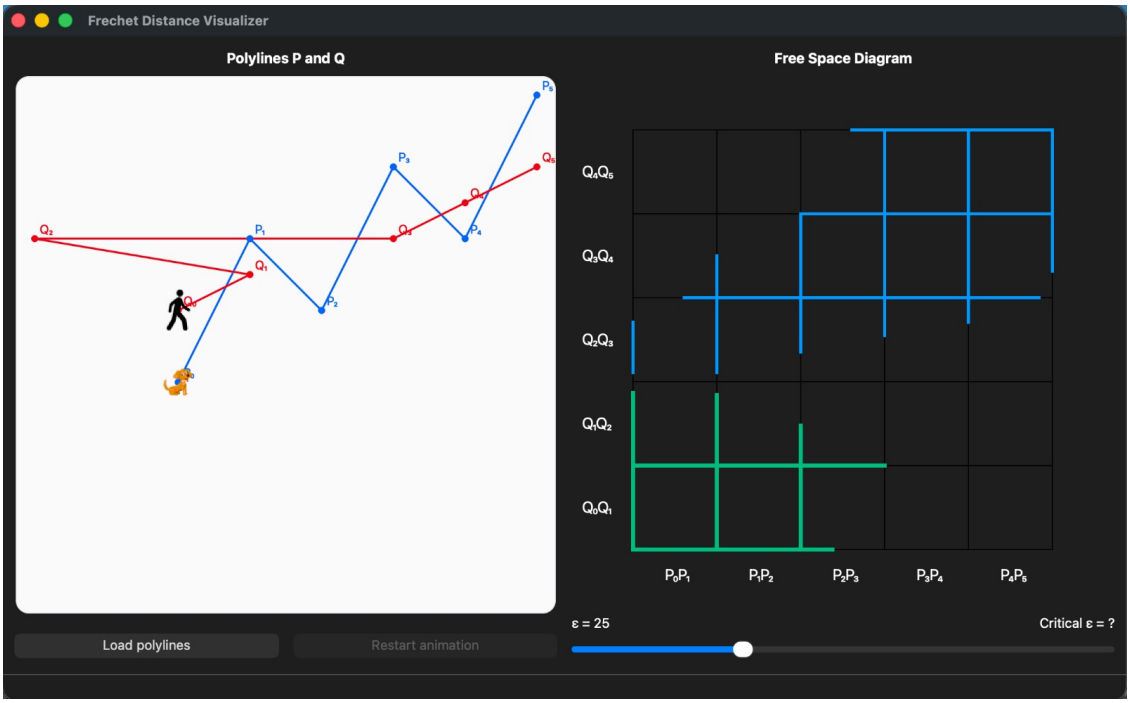


Figure 1: Application interface - before the critical leash length is calculated.

## 4 Implementation

### 4.1 System architecture

The application is implemented in C++ using the Qt framework [3]. It consists of three main components:

- **PolylineCanvas**, responsible for displaying the input curves and the animation of the traversal.
- **FreeSpaceCanvas**, which visualizes the free space diagram, reachable regions and the possible traversal path.
- **FreeSpace** and **Reachability** classes, which implement the geometric computations.

The user interface allows interactive control over the distance threshold $\varepsilon$ and supports loading curve data from external files.

### 4.2 Computing the free space

For each pair of segments $P_iP_{i+1}$ from $P$ and $Q_jQ_{j+1}$ from $Q$, the free space is defined as the set of parameter pairs $(s,t)$ within the cell $[i,i+1]\times[j,j+1]$ satisfying

$$\|P(s)-Q(t)\|\le\varepsilon,$$

where $s\in[i,i+1]$ and $t\in[j,j+1]$ [1].

Using linear interpolation along segments,

$$P(s)=(1-(s-i))P_i+(s-i)P_{i+1},\quad Q(t)=(1-(t-j))Q_j+(t-j)Q_{j+1}.$$

Plugging these expressions into the distance constraint gives

$$\|(P_i-Q_j)+(s-i)(P_{i+1}-P_i)-(t-j)(Q_{j+1}-Q_j)\|\le\varepsilon,$$

which is equivalent to

$$\big\|(P_i-Q_j)+s\,(P_{i+1}-P_i)-t\,(Q_{j+1}-Q_j)-i(P_{i+1}-P_i)+j(Q_{j+1}-Q_j)\big\|\le\varepsilon.$$

Expanding in component form in $\mathbb{R}^2$ gives a *quadratic inequality in $s$ and $t$*, which represents an ellipse (or degenerate ellipse) in the parameter plane. Therefore, each cell's free space is a convex region bounded by an ellipse. Convexity of the free space regions is the crucial assumption for the later determination of a monotonic non-decreasing traversal path.

For visualization, it is sufficient to compute the intersection of this ellipse with the cell boundaries, producing *free intervals on the top, bottom, left, and right edges* of the cell. In other words, it is sufficient to solve the distance inequality

$$\|P(s)-Q(t)\|\le\varepsilon$$

for the following four special cases corresponding to the cell boundaries:

$$s=i,\quad s=i+1,\quad t=j,\quad t=j+1.$$

Solving these one-dimensional inequalities for each boundary gives the exact intervals along the edges where a monotone path could enter or exit the cell. These *free intervals* are then used in the dynamic programming propagation to determine reachability across the grid.

### 4.3 Propagation of reachable intervals

Reachability is computed iteratively using dynamic programming across the grid of cells formed by the polygonal segments of polylines $P$ and $Q$. Each cell $(i,j)$ corresponds to the parameter intervals $s\in[i,i+1]$ along $P$ and $t\in[j,j+1]$ along $Q$.

Let $F_\varepsilon^{i,j}\subset[i,i+1]\times[j,j+1]$ denote the *free space* in cell $(i,j)$ for a given $\varepsilon$. For each cell, we define *reachable intervals* along the four boundaries:

$$R_{i,j}^{\text{left}}\subset[j,j+1],\quad R_{i,j}^{\text{bottom}}\subset[i,i+1],\quad R_{i,j}^{\text{right}}\subset[j,j+1],\quad R_{i,j}^{\text{top}}\subset[i,i+1].$$

Free space intervals are defined as the maximal continuous segments along the edges of a cell that lie entirely within the free space.

$$F_{i,j}^{\text{left}}\subset[j,j+1],\quad F_{i,j}^{\text{bottom}}\subset[i,i+1],\quad F_{i,j}^{\text{right}}\subset[j,j+1],\quad F_{i,j}^{\text{top}}\subset[i,i+1].$$

It is evident that the following inclusions hold:

$$R_{i,j}^{\text{left}}\subset F_{i,j}^{\text{left}},\quad R_{i,j}^{\text{bottom}}\subset F_{i,j}^{\text{bottom}},\quad R_{i,j}^{\text{right}}\subset F_{i,j}^{\text{right}},\quad R_{i,j}^{\text{top}}\subset F_{i,j}^{\text{top}}.$$

We will also denote the left and the right end of an interval $I$ by $I.l$ and $I.r$, respectively.

**Initialization.** The bottom-left cell $(0, 0)$ is initialized based on the free space at the starting point:

$$R_{0,0}^{\text{bottom}} = \begin{cases} F_{0,0}^{\text{bottom}}, \text{ if } \{0\} \in F_{0,0}^{\text{bottom}} \\ \emptyset, \text{ otherwise.} \end{cases} \qquad R_{0,0}^{\text{left}} = \begin{cases} F_{0,0}^{\text{left}}, \text{ if } \{0\} \in F_{0,0}^{\text{left}} \\ \emptyset, \text{ otherwise.} \end{cases}$$

For the remaining cells in the first row $(i, 0)$ with $i > 0$, the bottom boundary is reachable *only if* both the bottom reachable interval of the left cell, $R_{i-1,0}^{\text{bottom}}$, and the bottom free space interval of the current cell, $F_{i,0}^{\text{bottom}}$, contain i:

$$R_{i,0}^{\text{bottom}} = \begin{cases} F_{i,0}^{\text{bottom}}, \text{ if } \{i\} = R_{i-1,0}^{\text{bottom}} \cap F_{i,0}^{\text{bottom}} \\ \emptyset, \text{ otherwise.} \end{cases}$$

Similarly, for the remaining cells in the first column $(0, j)$ with $j > 0$, the left boundary is reachable *only if* both the left reachable interval of the cell below it, $R_{0,j-1}^{\text{left}}$, and the left free space interval of the current cell, $F_{0,j}^{\text{left}}$, contain j:

$$R_{0,j}^{\text{left}} = \begin{cases} F_{0,j}^{\text{left}}, \text{ if } \{j\} = R_{0,j-1}^{\text{left}} \cap F_{0,j}^{\text{left}} \\ \emptyset, \text{ otherwise.} \end{cases}$$

This ensures that reachability propagates correctly from the origin, and that unreachable regions along the first row or column remain empty, preventing illegitimate paths.

**Propagation.** For each cell $(i, j)$, the reachable intervals along the right and top boundaries are computed from the left and bottom reachable intervals. These formulas rely heavily on the convexity of the free space within the cell. To calculate the top and right reachable intervals, the bottom and left intervals must be known beforehand. Therefore, the order in which cells are processed is important. After initialization, computation begins at the bottom-left corner, proceeding row by row from bottom to top, and the following formulas are applied to each cell:

$$R_{i,j}^{\text{top}} = \begin{cases} F_{i,j}^{\text{top}}, \text{ if } R_{i,j}^{\text{left}} \neq \emptyset \\ [max(R_{i,j}^{\text{bottom}}.l, F_{i,j}^{\text{top}}.l), F_{i,j}^{\text{top}}.r], \text{ if } max(R_{i,j}^{\text{bottom}}.l, F_{i,j}^{\text{top}}.l) \leq F_{i,j}^{\text{top}}.r \\ \emptyset, \text{ otherwise.} \end{cases}$$

$$R_{i,j}^{\text{right}} = \begin{cases} F_{i,j}^{\text{right}}, \text{ if } R_{i,j}^{\text{bottom}} \neq \emptyset \\ [max(R_{i,j}^{\text{left}}.l, F_{i,j}^{\text{right}}.l), F_{i,j}^{\text{right}}.r], \text{ if } max(R_{i,j}^{\text{left}}.l, F_{i,j}^{\text{right}}.l) \leq F_{i,j}^{\text{right}}.r \\ \emptyset, \text{ otherwise.} \end{cases}$$

These intervals are then propagated to adjacent cells:

$$R_{i+1,j}^{\text{left}} = R_{i,j}^{\text{right}}, \quad R_{i,j+1}^{\text{bottom}} = R_{i,j}^{\text{top}}.$$

By iterating over all cells in row-major order, one determines which portions of the top and right boundaries are reachable. If the top-right corner $(m, n)$ lies within a reachable interval, a monotone path exists for the given $\varepsilon$, and the Fréchet distance can be updated or a critical path extracted.

**Remarks.** This propagation ensures that only *monotone paths* are considered, which is essential for computing the continuous Fréchet distance. Each reachable interval is convex due to the convexity of the free space within a cell, making the computation efficient. Moreover, the propagation can be implemented using simple interval operations, and the intersection with cell boundaries guarantees correctness of the dynamic programming procedure.

### 4.4 Critical path extraction

Once the top-right corner becomes reachable for the first time, the algorithm extracts a *critical path* through the free space diagram. This path represents an optimal traversal realizing the Fréchet distance.

The path is stored as a sequence of parameter pairs $(s, t)$ and is used directly for animation purposes. It is denoted in yellow on the free space diagram. (Fig. 2).

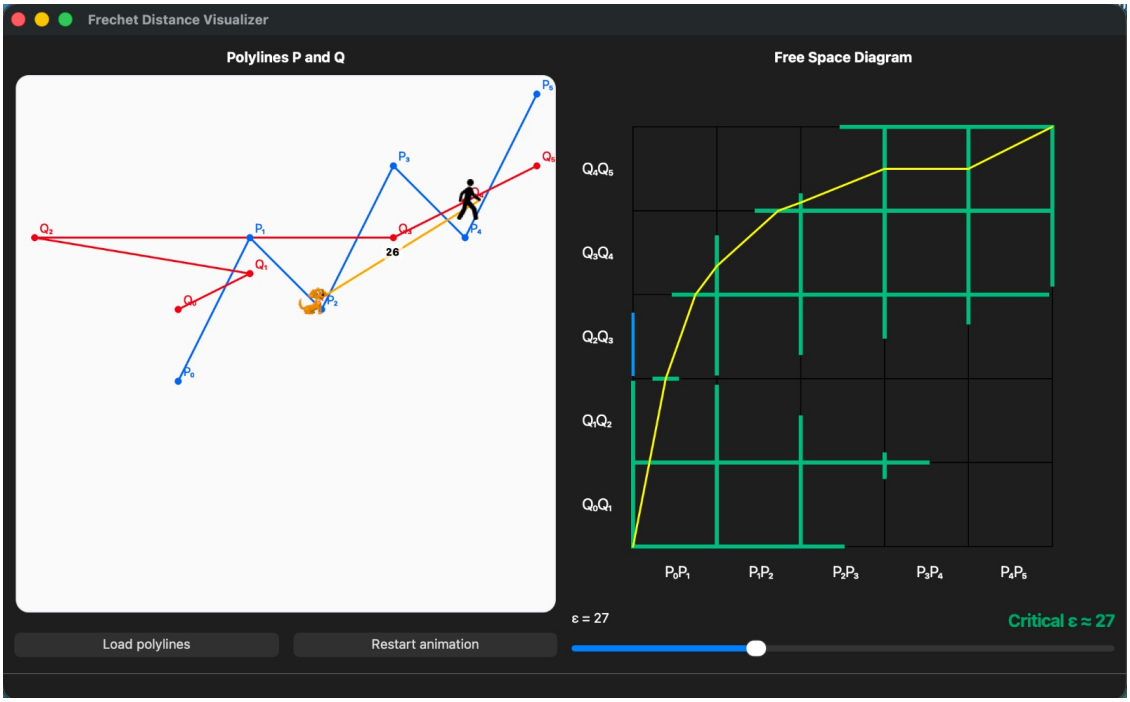


Figure 2: Application interface - after the critical leash length is calculated. Animation is in progress.

### 4.5 Complexity Analysis

Let $m$ and $n$ denote the number of segments of polylines $P$ and $Q$, respectively. The free space diagram consists of $m \times n$ cells.

For each cell, the free space intervals on the four boundaries are computed by solving a constant number of quadratic inequalities, resulting in $O(1)$ work per cell. Thus, the construction of the free space diagram takes $O(mn)$ time.

Reachability propagation is performed using dynamic programming over the grid, where each cell processes a constant number of interval operations. This phase also requires $O(mn)$ time.

If a monotone path exists, the critical path is reconstructed in linear time with respect to the grid dimensions - $O(m + n)$, which is dominated by the previous steps.

Overall, the time complexity of the algorithm is $O(mn)$, and the space complexity is also $O(mn)$ due to storing free space and reachable intervals.

## 5 Visualization and interaction

The left side of the interface displays the polygonal curves along with an animated traversal. Two icons represent the moving points on each curve, and a dynamic line segment illustrates the leash between them. The length of this segment is displayed and updated continuously during the animation.

The right side of the interface shows the free space diagram, including free intervals, reachable regions, and the critical path.

Users can load custom curve files, adjust the distance threshold using a slider whose range is derived from the curves' bounding box, and restart the animation when desired.

## 6 Conclusion and future work

This project demonstrates how free space diagrams can be used to effectively visualize and understand the Fréchet distance between polygonal curves. By combining geometric computation with interactive visualization, the tool provides both educational and exploratory value.

Possible extensions include support for discrete Fréchet distance, three-dimensional curves, performance optimizations, and exporting animations for presentation purposes.

## References


[1] Helmut Alt and Michael Godau. Computing the fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5(1–2):75–91, 1995.

[2] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3rd edition, 2008.

[3] The Qt Company. *Qt 6 Documentation: Painting and Graphics*, 2024. https://doc.qt.io/qt-6/painting.html.