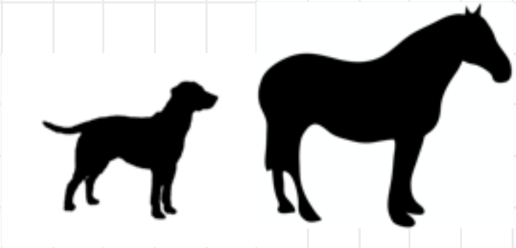
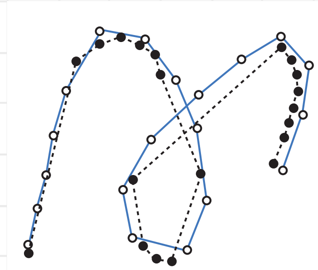
How to measure the distance/similarity between two sets/objects in the plane

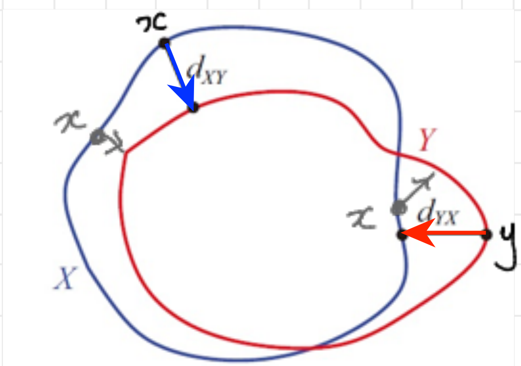




Don Sheehy

Applications:
- hand-writing, signatures
- protein backbones
- cartography

Hausdorff distance



GC https://structseg2019.grand-challenge.org/Evaluation/

$$d_{XY} = \max_{x \in X} \min_{y \in Y} d(x,y)$$

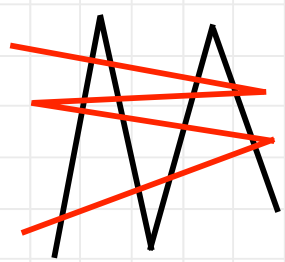
$$d_{YX} = \max_{y \in Y} \min_{x \in X} d(x,y)$$

Hausdorff distance = $\max\{d_{XY}, d_{YX}\}$

$d_{XY} = 0$

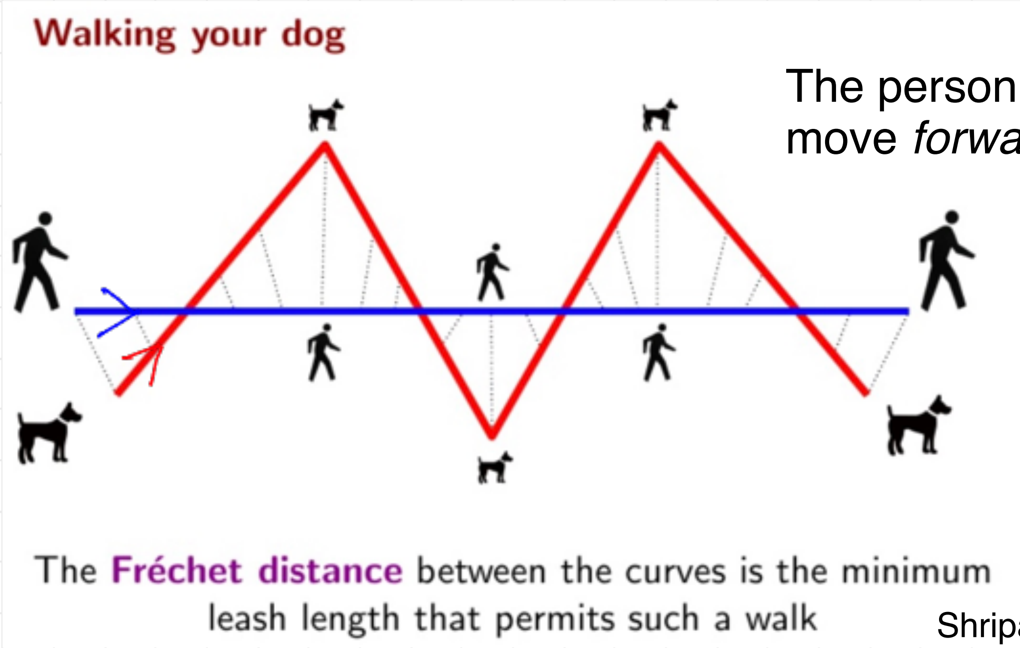
$d_{YX} > 0$

Hausdorff distance can be a bad measure for curves



every point is close to the other curve, but the curves are not similar (a curve is more than a set of points!)

A better distance measure for curves: Fréchet distance



The person and the dog must move *forward* on their curves.

Shripad Thite

A ***curve*** is a continuous map $[0,1] \longrightarrow R^2$ (map time [0,1] to points along the curve)
There can be many different parameterizations (corresponding to different speeds).

Definition. The **Fréchet distance** of two curves A and B is

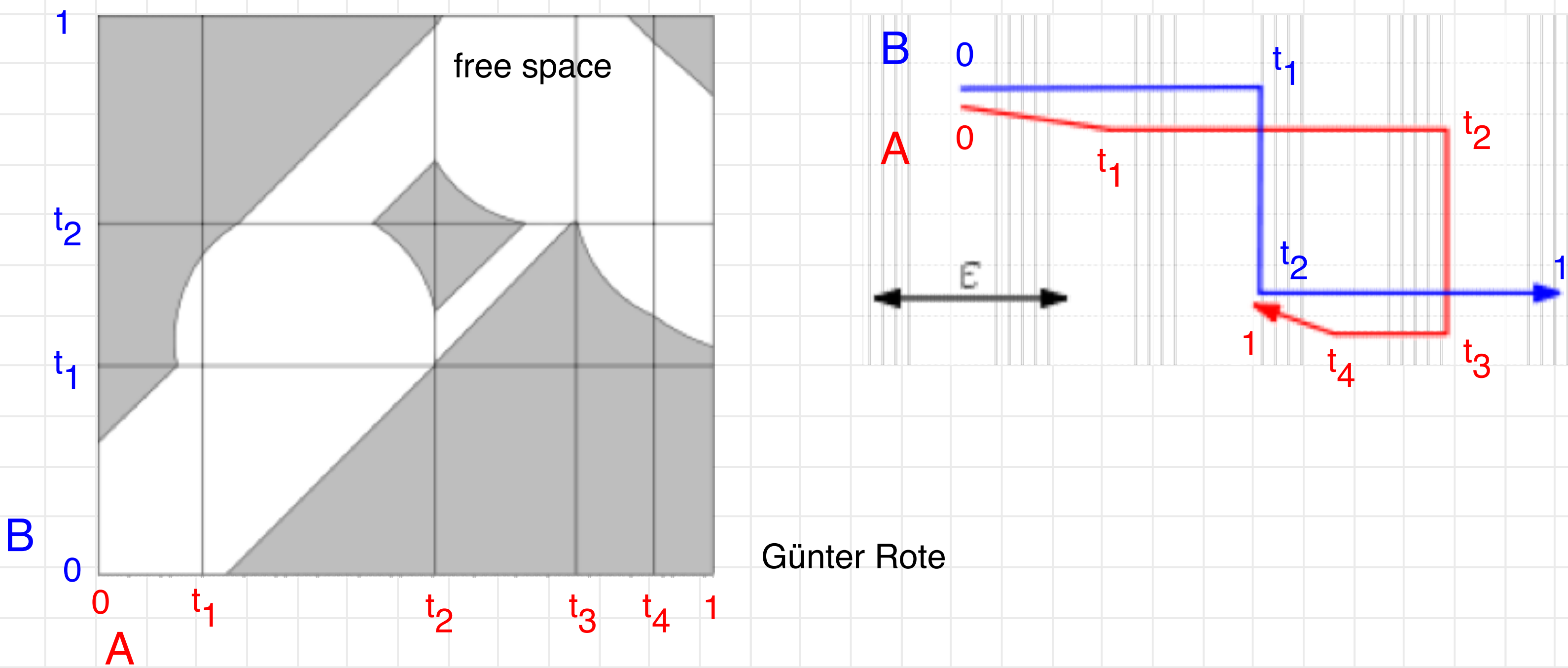
$$\min \max_{t \in [0,1]} \{ \underbrace{d(\alpha(t), \beta(t))}_{\text{this is the length of the leash at time } t} \}$$

reparameterizations
α of A and β of B

Algorithm for Fréchet distance between two polygonal curves in the plane
Alt and Godeau, 1995 https://doi.org/10.1142/S0218195995000064
The algorithm has two steps:

1. a decision procedure to see if the distance is $\leq \varepsilon$
2. a search to find the min ε

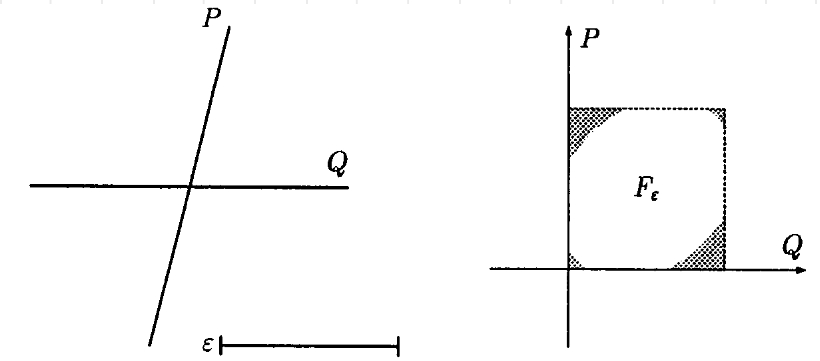
Step 1. Testing if the Fréchet distance is $\leq \varepsilon$
use the *free-space diagram* : points (t_a, t_b) such that $d(\alpha(t_a), \beta(t_b)) \leq \varepsilon$

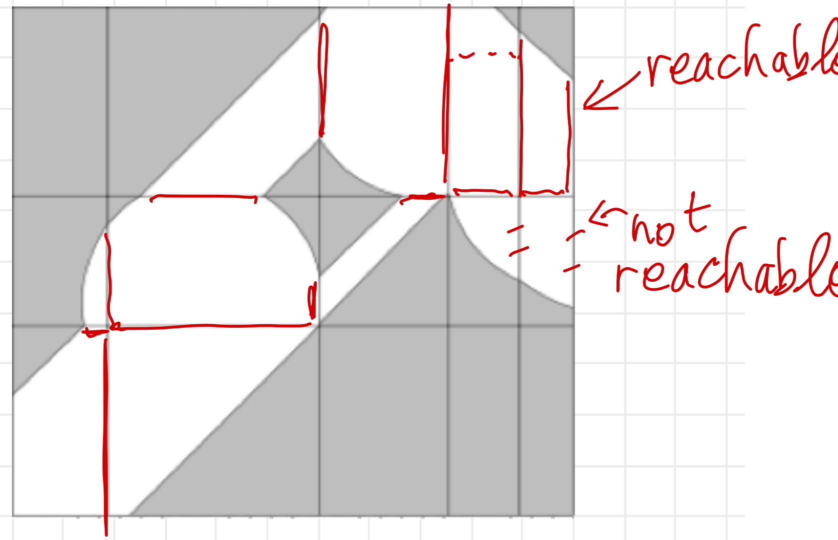


Günter Rote

Lemma. The Fréchet distance is $\leq \varepsilon$ iff there is path from (0,0) to (1,1) in the free space that is monotone in t_a and t_b.

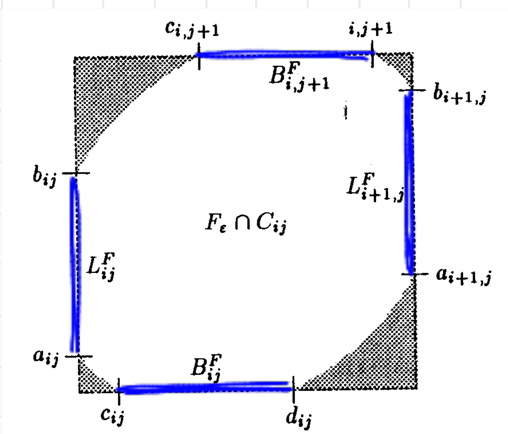
Step 1. Testing if the Fréchet distance is ≤ ε
How to construct the free space:



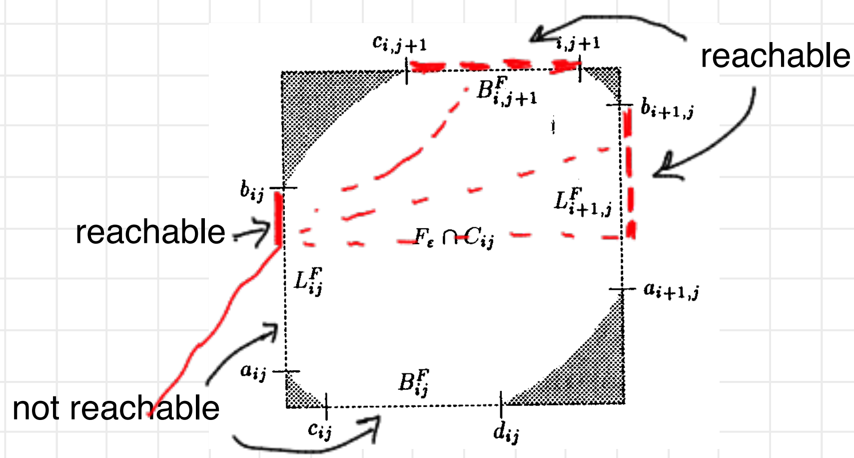


Lemma. For two line segments, the free space is [part of] an ellipse (possibly degenerating to a strip if the line segments are parallel).

We can compute the intervals of the free space along each grid line.



Then we can compute the subintervals reachable from (0,0) via a monotone path.



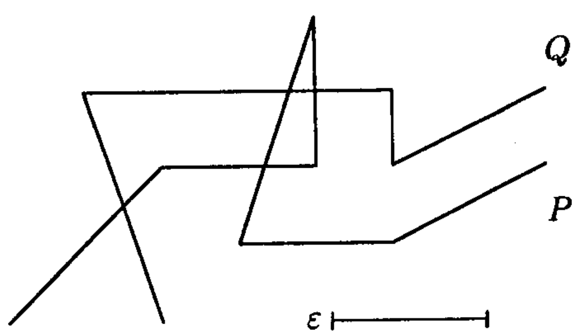
Algorithm for Fréchet distance between two polygonal curves in the plane

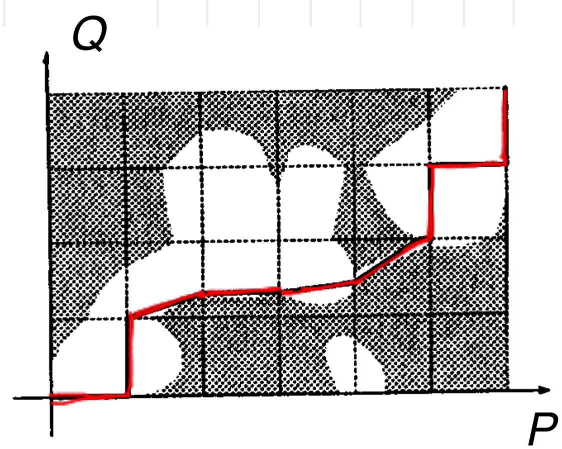
The algorithm has two steps:

1. a decision procedure to see if the distance is ≤ ε
2. a search to find the min ε

From the above, Step 1 can be done in time O(nm) for two polygonal curves with n and m edges, respectively.

another example

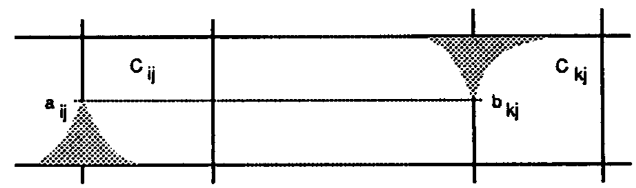




Step 2. Finding the minimum ε

Note that the free space grows as ε increases.
At ***critical*** values of ε there are significant changes to the free space:

1. when $\varepsilon = d(\alpha(0), \beta(0))$ — then the free space contains (0,0)
2. when $\varepsilon = d(\alpha(1), \beta(1))$ — then the free space contains (1,1)
3. when an interval of free space opens up between two cells
4. when a new horizontal or vertical passage opens up

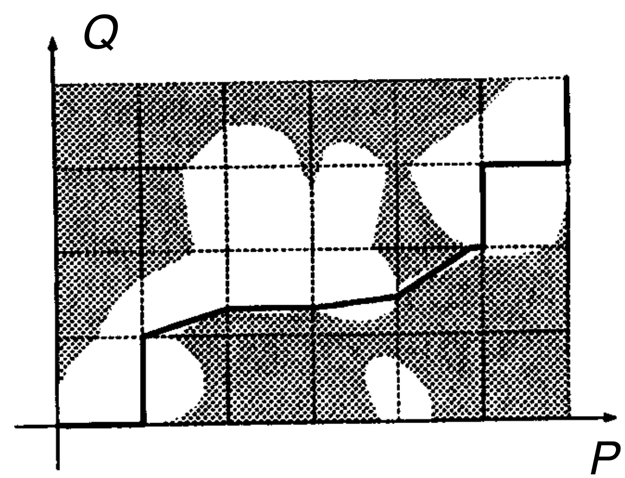


a horizontal passage opens

(1) and (2) are single distances

(3) is the distance between a vertex of one curve and an edge of the other curve, so $O(nm)$ events.

(4) involves two vertices of one curve and an edge of the other curve, so $O(n^2m + nm^2)$ events.



These events can be found in time $O(1)$ each.

Sort the events $O((n^2m + nm^2) \log (nm))$.
Do binary search to find minimum ε.

Total time: $O((n^2m + nm^2) \log (nm))$

Algorithm for Fréchet distance between two polygonal curves in the plane

The algorithm has two steps:

1. a decision procedure to see if the distance is $\leq \varepsilon$
2. a search to find the min ε

From above, Step 2 can be done in time $O((n^2m + nm^2) \log (nm))$.
It can be improved to $O(nm \log(nm))$ using "parametric search", a technique due to Megiddo.
We can write this bound as $O(n^2 \log n)$ where n is total input size.

A lower bound for Fréchet distance

There is no subquadratic algorithm assuming the Strong Exponential Time Hypothesis (SETH)

SETH says that 3-SAT has no subexponential time algorithm. This is stronger than assuming P ≠ NP. W https://en.wikipedia.org/wiki/Exponential_time_hypothesis

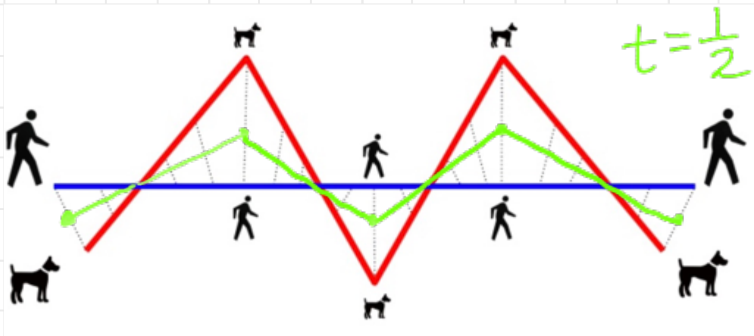
Karl Bringmann. "Why walking the dog takes time: Frechet distance has no strongly subquadratic algorithms unless SETH fails." In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science* d 10.1109/FOCS.2014.76

Fréchet distance in practice

Karl Bringmann, Marvin Künnemann, and André Nusser. "Walking the Dog Fast in Practice: Algorithm Engineering of the Fréchet Distance." In *35th International Symposium on Computational Geometry (SoCG 2019)* d 10.4230/LIPIcs.SoCG.2019.17

Variants on Fréchet distance

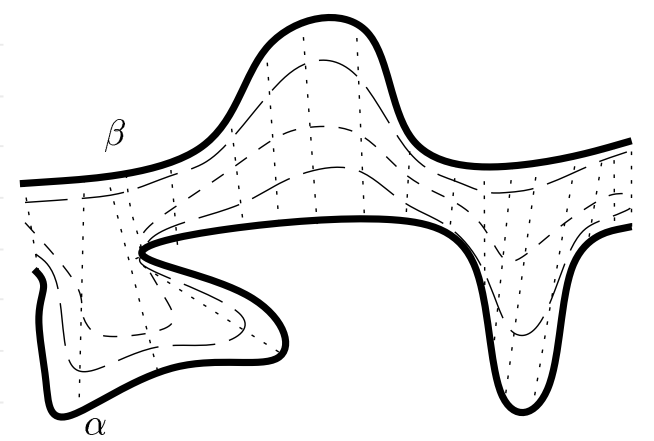
the leashes provide a way of morphing one curve to another



but the intermediate curves might not be simple even if the originals are

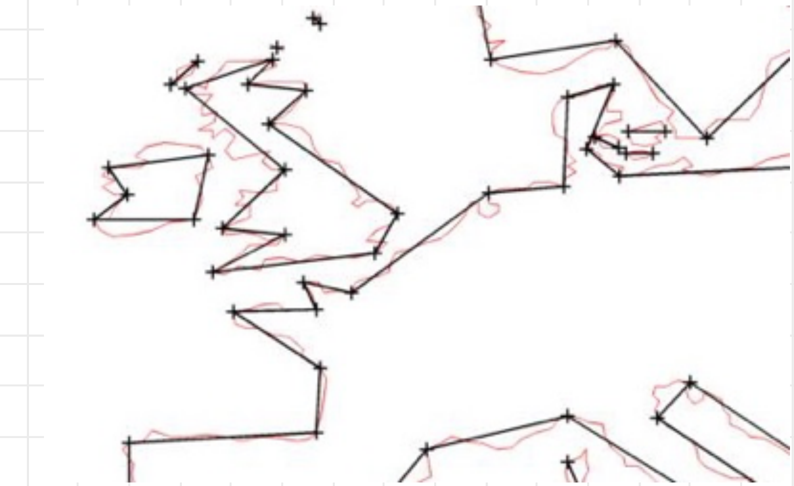
Efrat, Alon, Leonidas J. Guibas, Sariel Har-Peled, Joseph SB Mitchell, and T. M. Murali. "New Similarity Measures between Polylines with Applications to Morphing and Polygon Sweeping." (2002).

https://doi.org/10.1007/s00454-002-2886-1



when the leash must stay inside the region bounded by the input curves, the intermediate curves are simple

Polyline simplification.
Given a polyline, delete points while keeping the curve "close" to the original.
Important in cartography.



https://www.mathworks.com/matlabcentral/fileexchange/21132-line-simplification

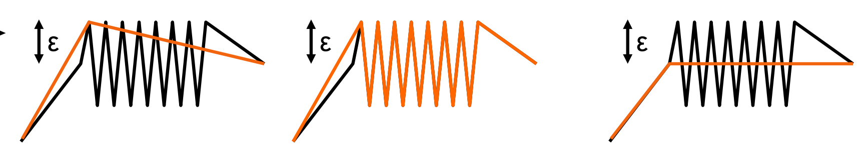
Douglas–Peucker Algorithm [1973]
Input: $p_1, \ldots, p_n$, error ε

call Test(1, n)

Test (i, j)
If all points p_k, $i < k < j$ are within distance ε of line segment $p_i p_j$, then delete all p_k
Else
let p_k be the farthest point
Test(i,k), Test(k,j)

David Legland, Marie-Françoise Devaux, Fabienne Guillon

Properties:
- output has Hausdorff distance $\leq \varepsilon$
- may not give the min number of points →
- may not keep the curve simple



Douglas-Peucker Optimal

Kevin Buchin

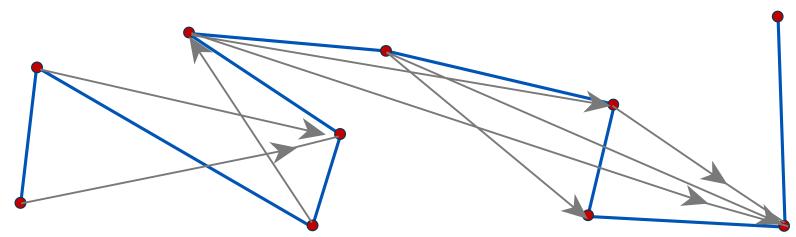
Runtime $O(n^2)$, can be improved

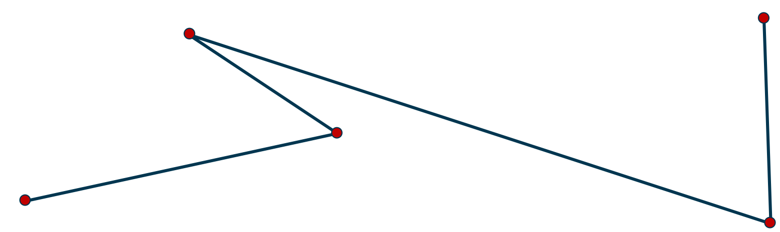
Polyline simplification.
Given a polyline, delete points while keeping the curve “close” to the original.

Imai-Iri Algorithm [1988]
Input: $p_1, \ldots, p_n$, error ε

Construct graph G with edge (i, j) if all points p_k, $i < k < j$ are within distance ε of line segment $p_i\, p_j$

Find a shortest path from 1 to n

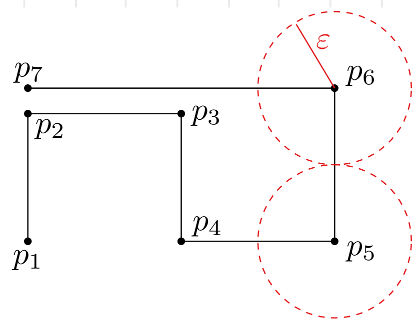




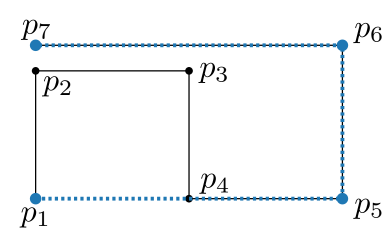
Kevin Buchin

Properties:
- output has Hausdorff distance ≤ ε
- may not give the min number of points
- does not always keep the curve simple

Runtime $O(n^3)$ can be improved to $O(n^2)$



both algorithms leave this curve intact



this smaller simplification has Hausdorff distance ε

http://dx.doi.org/10.4230/LIPIcs.SoCG.2018.56

Polyline simplification.
Given a polyline, delete points while keeping the curve “close” to the original.

Using the Fréchet distance rather than Hausdorff

Marc van Kreveld, Maarten Löffler, and Lionov Wiratma. "On Optimal Polyline Simplification using the Hausdorff and Fréchet Distance."

http://dx.doi.org/10.4230/LIPIcs.SoCG.2018.56

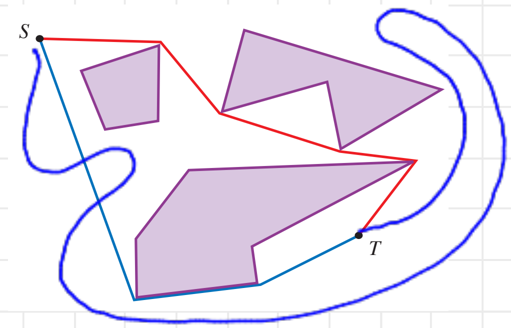
Further reading

Agarwal, Pankaj K., Sariel Har-Peled, Nabil H. Mustafa, and Yusu Wang. "Near-linear time approximation algorithms for curve simplification." *Algorithmica* 42, no. 3-4 (2005): 203-219.

d https://doi.org/10.1007/s00453-005-1165-y

Homotopic Paths

Two curves from s to t in the presence of obstacles are ***homotopic*** if one can be deformed to the other without intersecting the obstacles.



blue paths are homotopic

O(n log n) to test if two *simple* paths are homotopic

Sergio Cabello, Yuanxin Liu, Andrea Mantler, and Jack Snoeyink. "Testing homotopy for paths in the plane." *Discrete & Computational Geometry* 31, no. 1 (2004): 61-81.

d https://doi.org/10.1007/s00454-003-2949-y

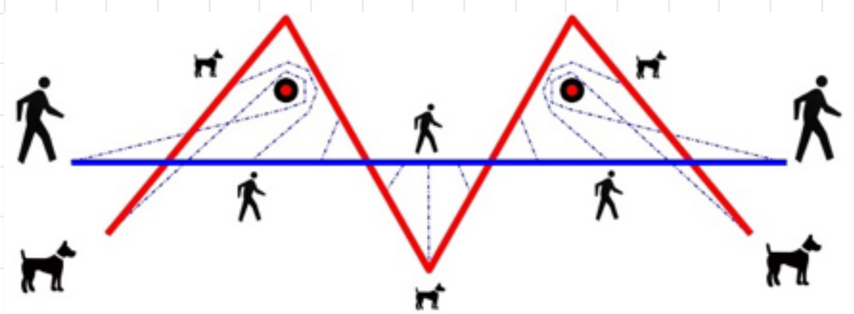
Finding a shortest path homotopic to a given one

Alon Efrat, Stephen G. Kobourov, and Anna Lubiw. "Computing homotopic shortest paths efficiently." *Computational Geometry* 35, no. 3 (2006): 162-172.

d https://doi.org/10.1016/j.comgeo.2006.03.003

Combining homotopic and Fréchet distance

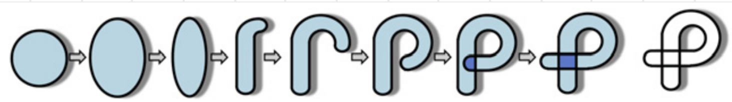
Chambers, Erin Wolf, Eric Colin De Verdiere, Jeff Erickson, Sylvain Lazard, Francis Lazarus, and Shripad Thite. "Homotopic Fréchet distance between curves or, walking your dog in the woods in polynomial time." *Computational Geometry* 43, no. 3 (2010): 295-311.



d https://doi.org/10.1016/j.comgeo.2009.02.008

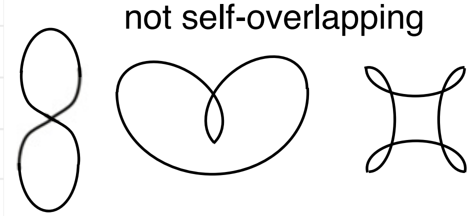
Self-Overlapping Curves

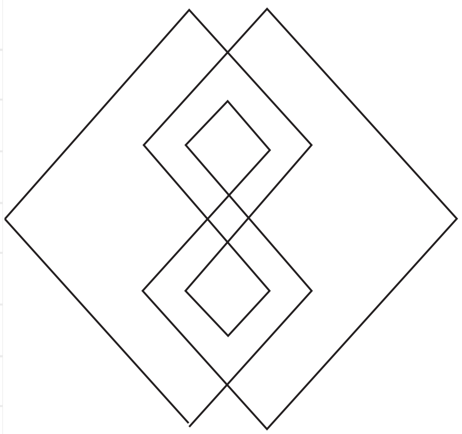
A self-overlapping curve is formed by stretching a disk. Overlapping is allowed. Twisting in 3D is not.

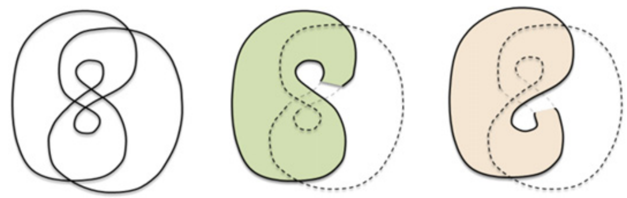


Uddipan Mukherjee

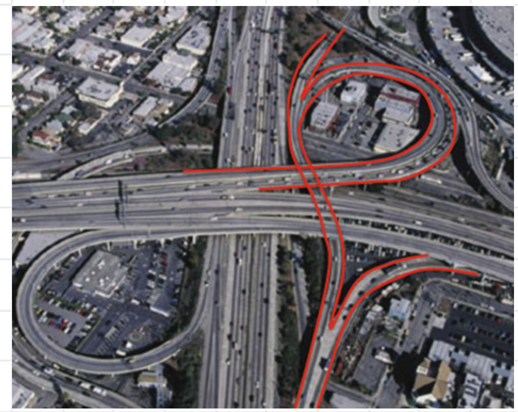
not self-overlapping







two different immersions of “Milnor’s doodle”



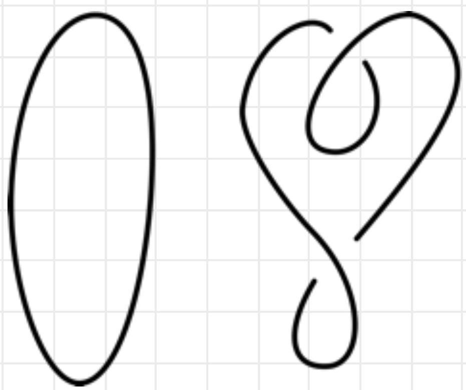
An $O(n^3)$ time dynamic programming algorithm to detect self-overlapping curves

Peter W. Shor, and Christopher J. Van Wyk. "Detecting and decomposing self-overlapping curves." *Computational Geometry* 2, no. 1 (1992): 31-50. https://doi.org/10.1016/0925-7721(92)90019-O

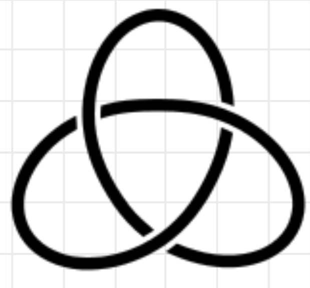
Evans, Parker, and Carola Wenk. "Combinatorial Properties of Self-Overlapping Curves and Interior Boundaries." (2020). arXiv:2003.13595

The unknot problem. https://en.wikipedia.org/wiki/Unknotting_problem

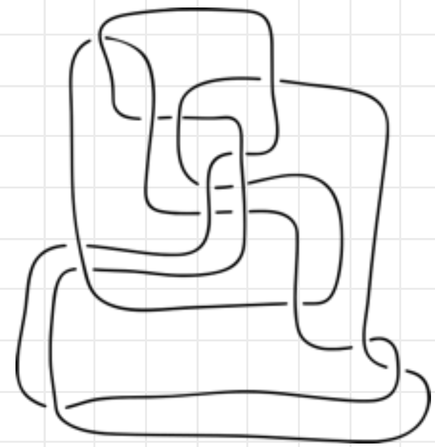
Given a knot diagram, does it represent the unknot?



two diagrams
of the unknot



the trefoil knot



is this the unknot?

1999. The unknot problem is in NP. 10.1145/301970.301971

2016. The unknot problem is in co-NP. https://arxiv.org/abs/1604.00290

OPEN. Is there a polynomial time algorithm for the unknot problem?

Summary

- curves
    - Fréchet distance
    - curve simplification
    - self-overlapping curves
    - knots

References

- on slides