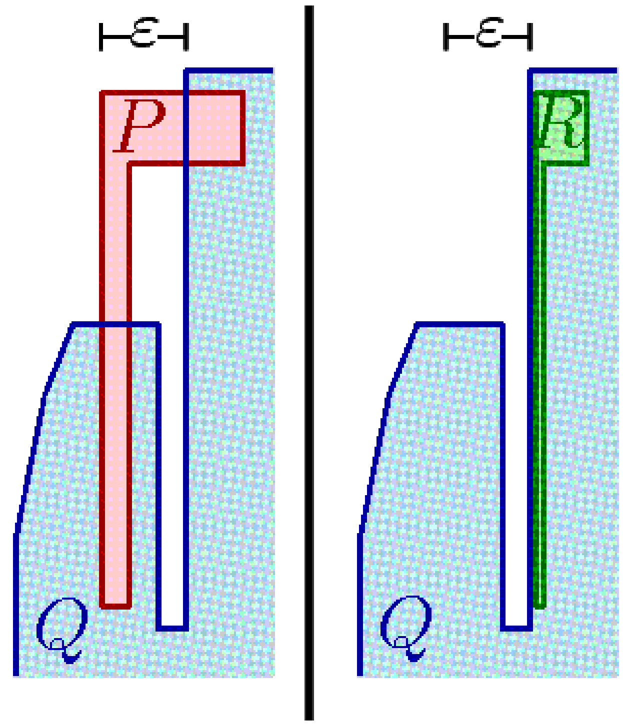
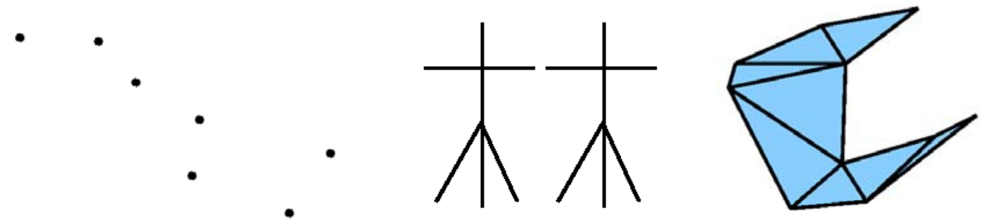
Partial Matching between Surfaces Using Fréchet Distance

Carola Wenk
University of Texas at San Antonio
Joint work with Jessica Sherette



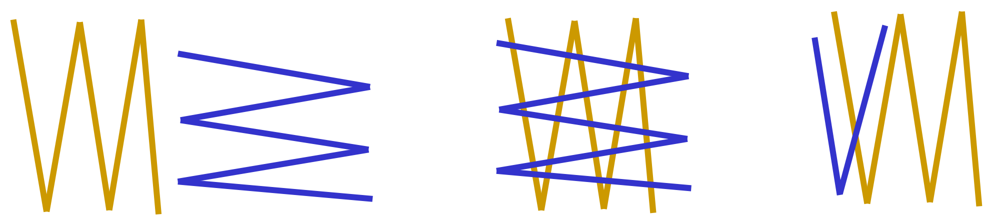
Geometric Shape Matching

- Consider **geometric shapes** to be composed of a number of basic objects such as



points line segments triangles

- How **similar** are two geometric shapes?



→
- Choice of distance measure
- Full or partial matching
- Exact or approximate matching
- Transformations (translations, rotations, scalings)

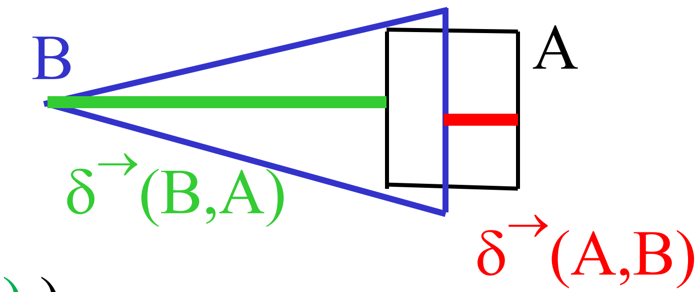
Shape Matching - Applications

- Character Recognition
- Fingerprint Identification
- Molecule Docking, Drug Design
- Image Interpretation and Segmentation
- Quality Control of Workpieces
- Robotics
- Pose Determination of Satellites
- Puzzling
- . . .

Distance Measures

- Directed Hausdorff distance

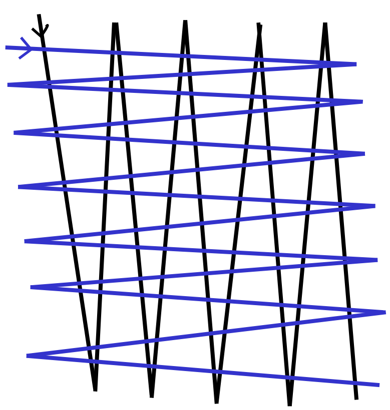
$$\vec{\delta}(A,B) = \max_{a \in A} \min_{b \in B} \| a-b \|$$



- Undirected Hausdorff-distance

$$\delta(A,B) = \max\left(\vec{\delta}(A,B)\ ,\ \vec{\delta}(B,A)\right)$$

But:



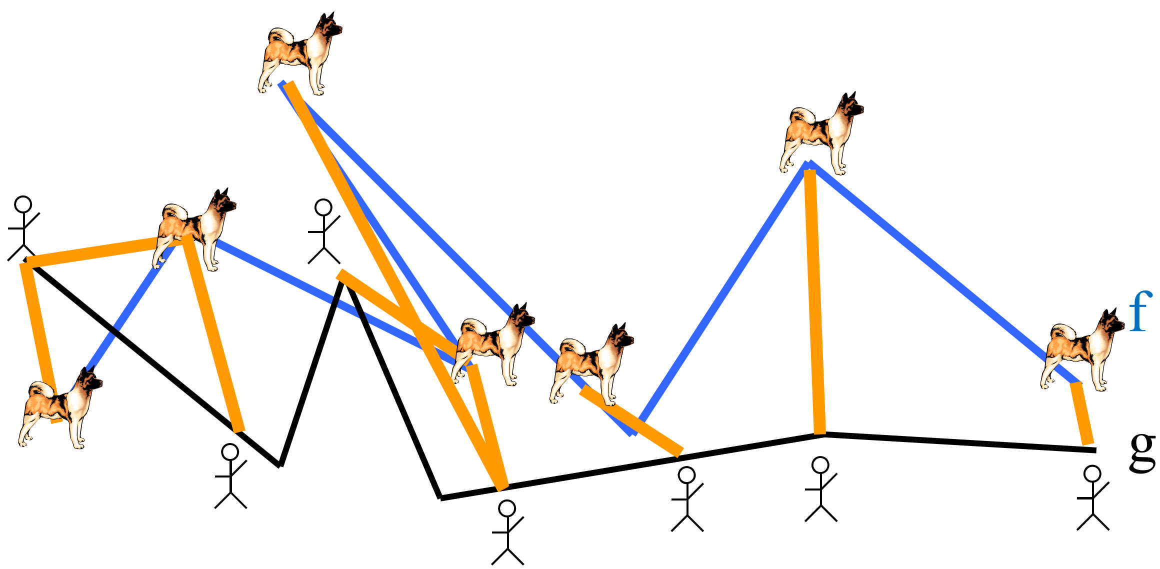
- Small Hausdorff distance
- When considered as curves the distance should be large
- The Fréchet distance is well-suited to compare continuous shapes.

Fréchet Distance for Curves

f,g: [0,1] $\rightarrow$ R^2

$$\delta_F(f,g) = \inf_{\sigma:[0,1]\rightarrow[0,1]} \max_{t\in[0,1]} \|f(t)-g(\sigma(t))\|$$

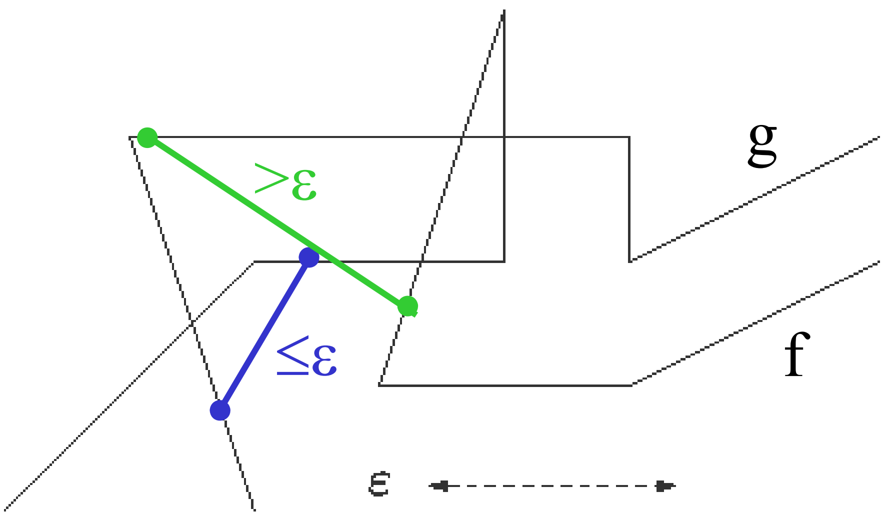
where α and β range over continuous monotone increasing reparameterizations only.

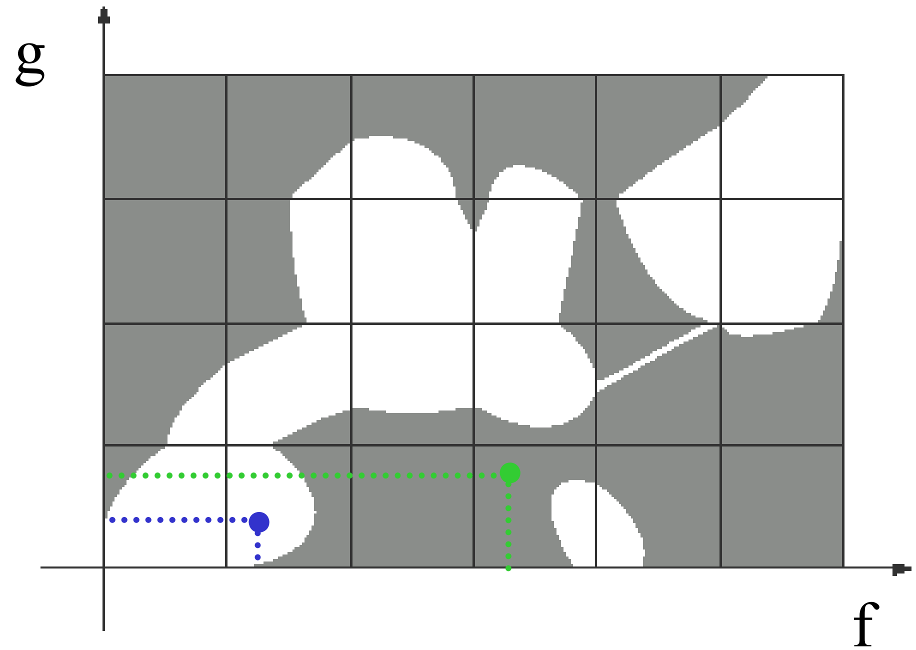


- Man and dog walk on one curve each
- They hold each other at a **leash**
- They are only allowed to go forward
- δ_F is the minimal possible leash length

[F06] M. Fréchet, Sur quelques points de calcul fonctionel, *Rendiconti del Circolo Mathematico di Palermo* 22: 1-74, 1906.

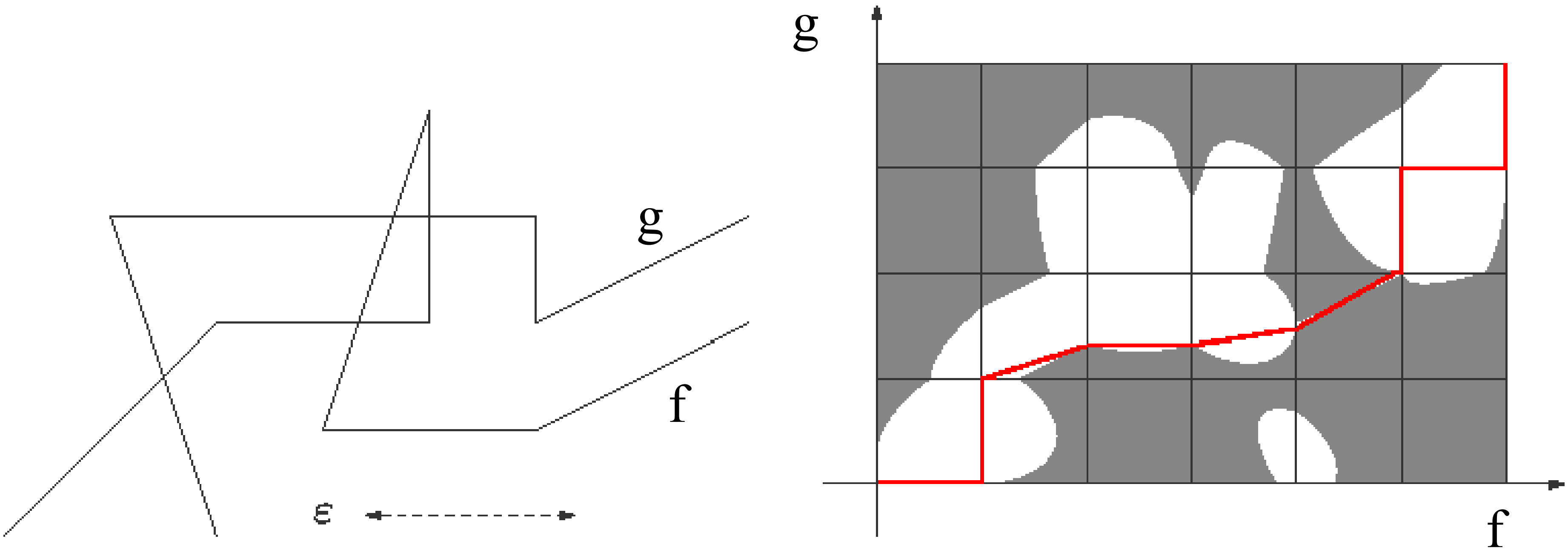
Free Space Diagram





- $F_\varepsilon(f,g) = \{ (s,t) \in [0,1]^2 \mid \| f(s) - g(t) \| \leq \varepsilon \}$ *white points*
  free space of f and g

Free Space Diagram

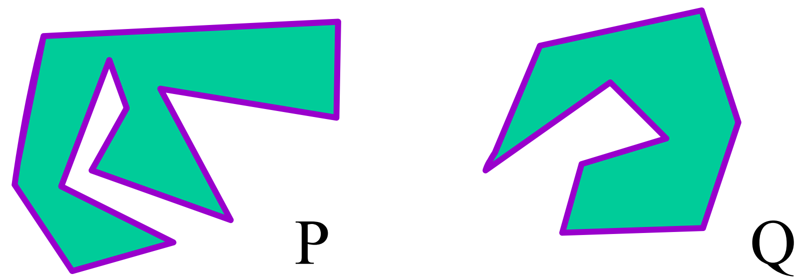


- $F_\varepsilon(f,g) = \{ (s,t) \in [0,1]^2 \mid \| f(s) - g(t) \| \leq \varepsilon \}$ *white points*
  free space of f and g
- $\delta_F(f,g) \leq \varepsilon$ iff there is a monotone path in the free space from (0,0) to (1,1)
- Can be decided using DP in O(mn) time [AG95]

[AG95] H. Alt, M. Godau, Computing the Fréchet distance between two polygonal curves, *IJCGA* 5: 75-91, 1995.

Fréchet Distance for Surfaces

- **Given:** Two surfaces
  $P,Q: [0,1]^2 \rightarrow R^d$

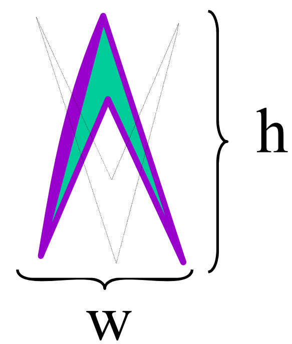


- The Fréchet distance is defined as:

$$\delta_F(P,Q) = \inf_{\substack{\sigma: P \rightarrow Q \\ \sigma \text{ homeomorphism}}} \max_{t \in P} \|t-\sigma(t)\|$$

- Is δ_F (**P**,**Q**) = δ_F (**∂P**, **∂Q**)?

  No: δ_F (**∂P**, **∂Q**) ≈ w/2
  δ_F (**P**,**Q**) ≈ h/2



[BBW08] K. Buchin, M. Buchin, C. Wenk, Computing the Fréchet Distance Between Simple Polygons, *CGTA* 41: 2-20, 2008.

Fréchet Distance for Surfaces

- **For piecewise linear surfaces:**
  - [G98] [BBS10] Computing δ_F is NP-hard, even when one surface is a triangle, or when both surfaces are polygons with holes or terrains
  - [AB09] δ_F is upper-semi-computable; it is unknown if it is computable
- **For simple polygons:**
  - [BBW08] δ_F can be computed in polynomial time
  - [SW12] Partial δ_F can be decided in polynomial time
- **For folded polygons:**
  - [CDHS**W**11] δ_F can be approximated in polynomial time

[BBS10] K. Buchin, M. Buchin, A. Schulz, Fréchet distance for surfaces: Some simple hard cases, ESA: 63-74, 2010.

[SW12] J. Sherette, C. Wenk, Computing the Partial Fréchet Distance Between Polygons, SWAT, 2012.

[CDHSW11] A.F.Cook IV, A. Driemel, S. Har-Peled, J. Sherette, C. Wenk, Computing the Fréchet….Folded Polygons,WADS, 2011.

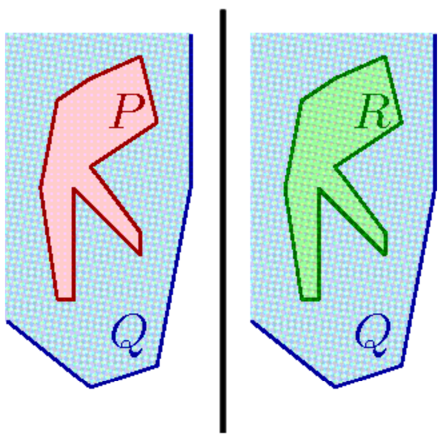
[BBW08] K. Buchin, M. Buchin, C. Wenk, Computing the Fréchet Distance Between Simple Polygons, *CGTA* 41: 2-20, 2008.

[G98] M. Godau, On the complexity of measuring the similarity…, Dissertation, Freie Universität Berlin, 1998.

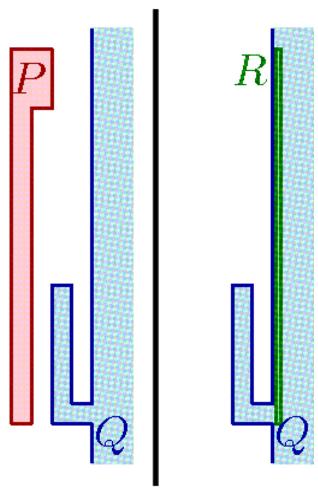
[AB09] H. Alt, M. Buchin, Can we compute the similarity between surfaces?, D&CG, to appear.

Partial Fréchet Distance

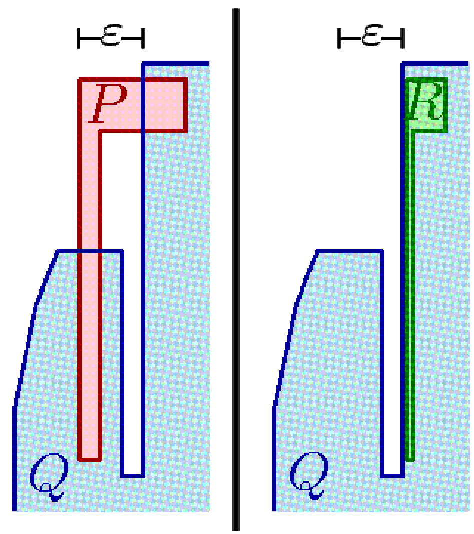
- **Given:** Two simple polygons P, Q (coplanar, triangulated), and some $\varepsilon>0$.
- **Task:** Decide whether there exists a simple polygon $R \subseteq Q$ such that $\delta_F(P,R) \le \varepsilon$.



- $P \subseteq Q$
$\Rightarrow R=P, \varepsilon=0$



- $P \cap Q = \emptyset$
$\Rightarrow$ Similar to projecting P to ∂Q

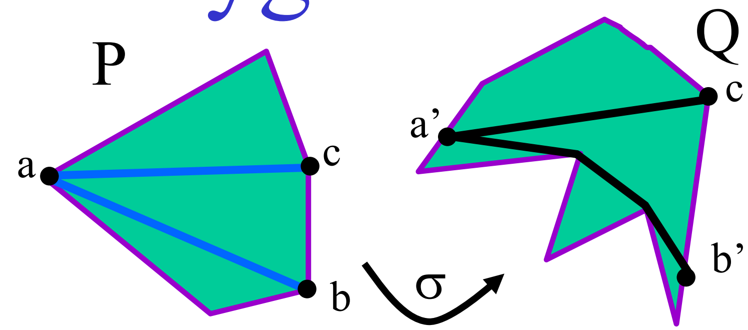


- Points in $P \cap Q$ are not mapped straight down
- Mapping of points is not independent from other points

Approach for Fréchet Distance between Simple Polygons

Restrict the homeomorphisms:

Map diagonals in P only to **shortest paths** in Q.



For $\varepsilon>0$, find homeom. such that:

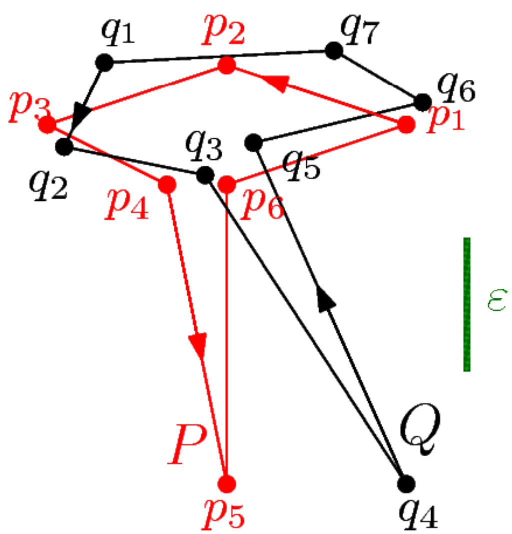
1. $\delta_F(\partial P, \partial Q) \leq \varepsilon$ (specifies mapping for diagonal endpoints)
2. Every diagonal D in P has distance $\leq \varepsilon$ to corresponding shortest path in Q

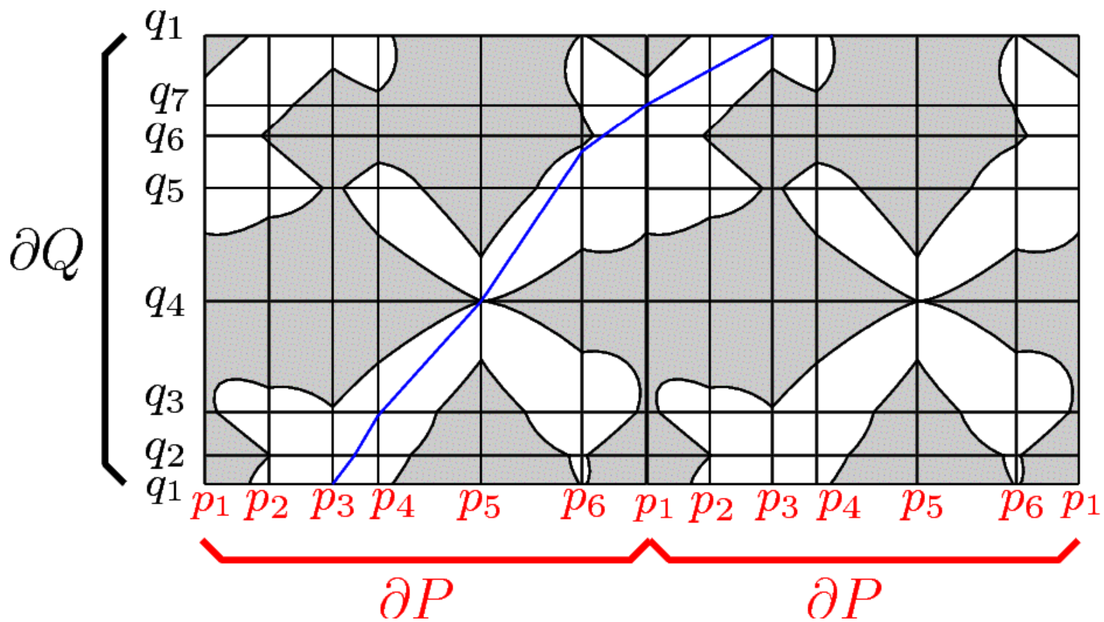
Map boundary & check diagonals:
Compute combinatorially equivalent mappings from ∂P to ∂Q, that also ensure small δ_F between diagonals and shortest paths

This yields a polynomial-time algorithm.

[BBW08] K. Buchin, M. Buchin, C. Wenk, Computing the Fréchet Distance Between Simple Polygons, *CGTA* 41: 2-20, 2008.

(Double) Free Space Diagram





- Free space diagram: ∂P×∂Q
- Boundary mapping from ∂P to ∂Q corresponds to a monotone path from bottom to top (that maps all of P).

Approach for Partial Fréchet Distance between Simple Polygons

- Since we have to find $R \subseteq Q$, the boundary of R is not known.
  $\Rightarrow$ Cannot just map boundaries anymore.
- We extend simple polygons approach in a different way:

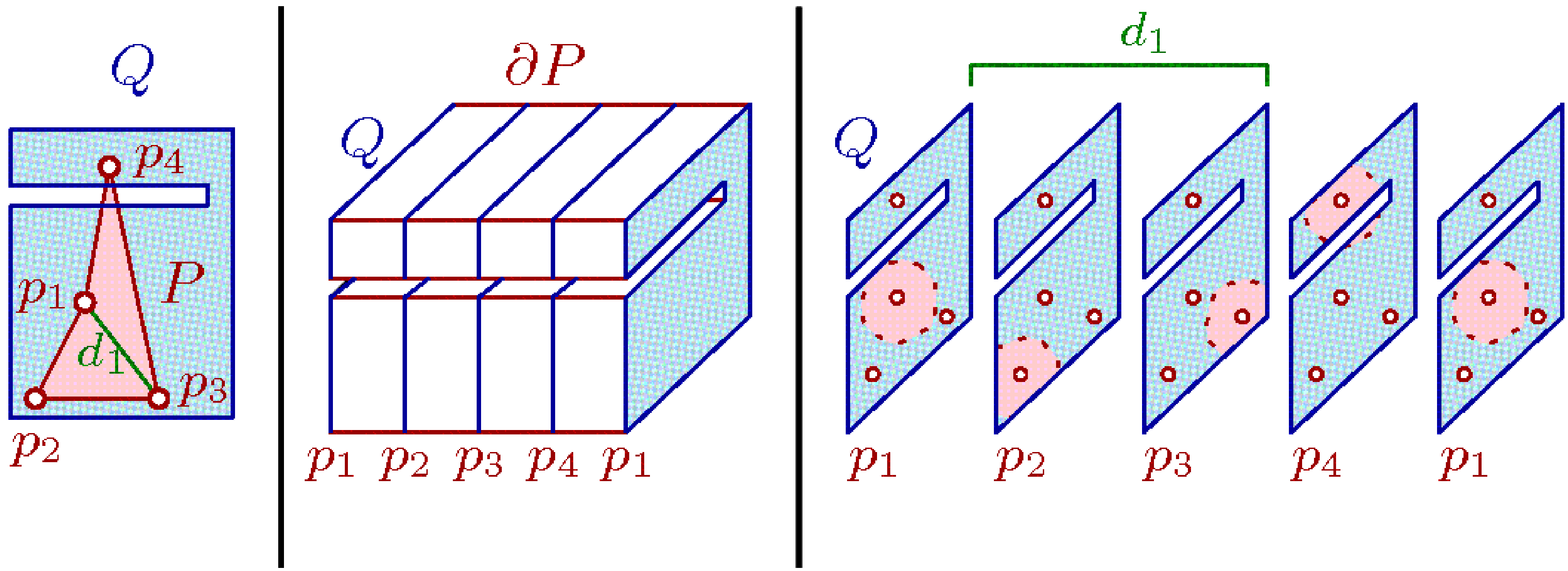
1. Map boundary & check diagonals:

Compute combinatorially equivalent mappings from ∂P to ~~∂Q~~ **some closed curve in Q**, that also ensure small δ_F between diagonals and shortest paths .
$\Rightarrow$ (Q,ε)-valid set of neighborhoods

2. Construct R from (Q,ε)-valid set

Prove that a (Q,ε)-valid set of neighborhoods always contains a valid **simple** polygon $R \subseteq Q$

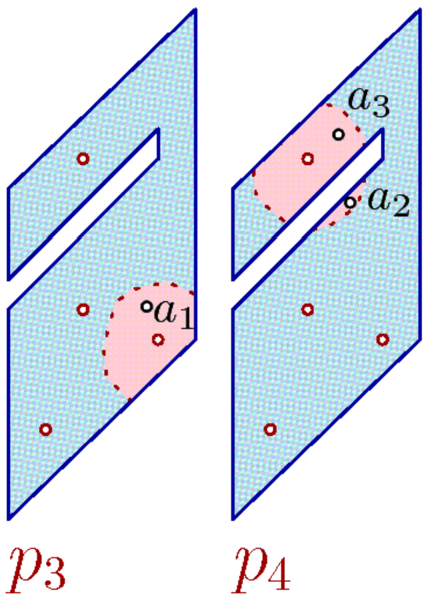
3D Free Space Diagram



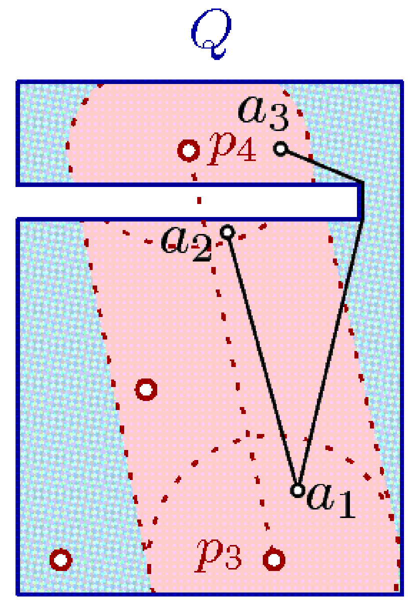
- Free space diagram: $\partial P \times Q$
- Sequence of **slices** $p_i \times Q$
- Boundary mapping from ∂P to closed curve in Q corresponds to a monotone path from first slice to last slice.
- Note: Path need only be monotone along P.

Reachability

Pair of adjacent slices in free space diagram:

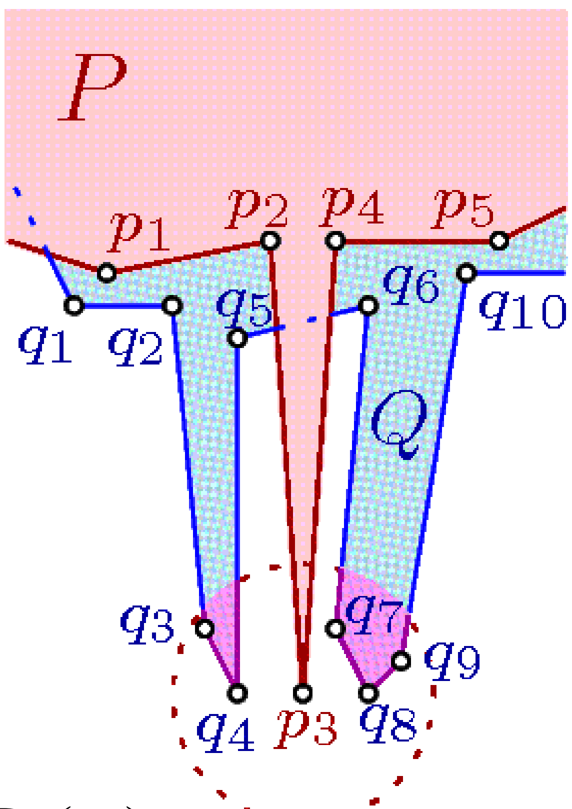


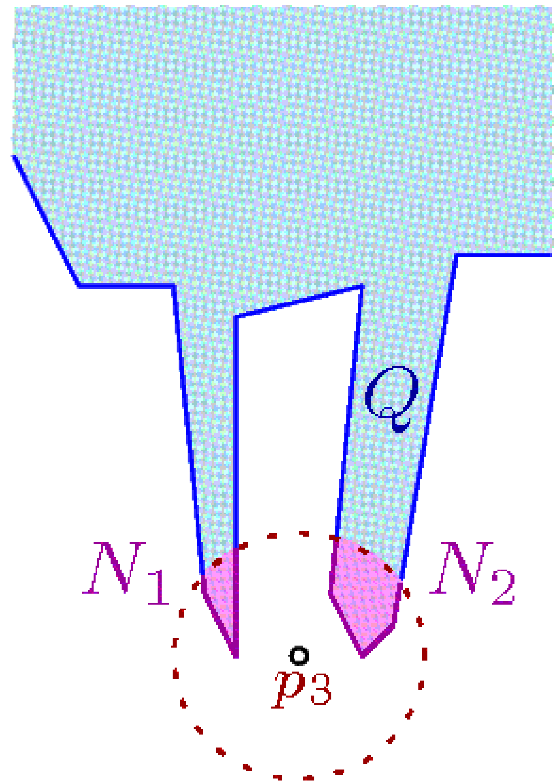
Scenario in Q:



- a_2 is **reachable** from a_1 iff $\delta_F(\overline{p_3 p_4}, \pi(a_1,a_2)) \le \varepsilon$, where $\pi(a_1,a_2)$ is the shortest path in Q between a_1 and a_2.
- a_2 is reachable from a_1, but a_3 is not.

Neighborhoods

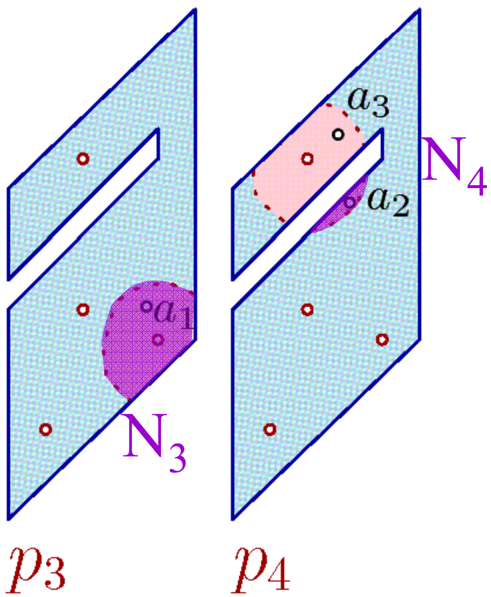




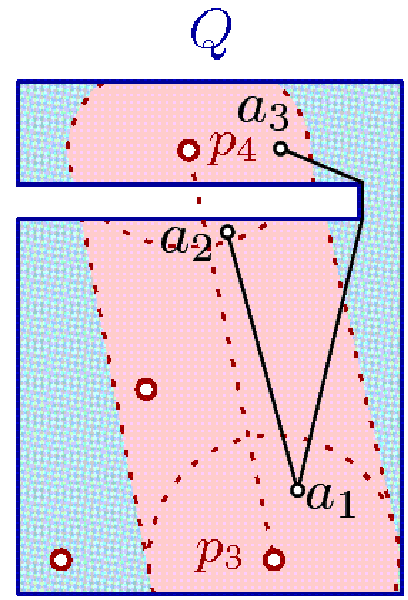
- ε-disk $D_\varepsilon(p_3)$. ⊢ ε ⊣
- Points in **$D_\varepsilon(p_3) \cap Q$** can be mapped to p_3.
- **Neighborhood** of p_i: Maximal connected subset of $D_\varepsilon(p_i) \cap Q$

Propagate Reachability

Pair of adjacent slices in free space diagram:



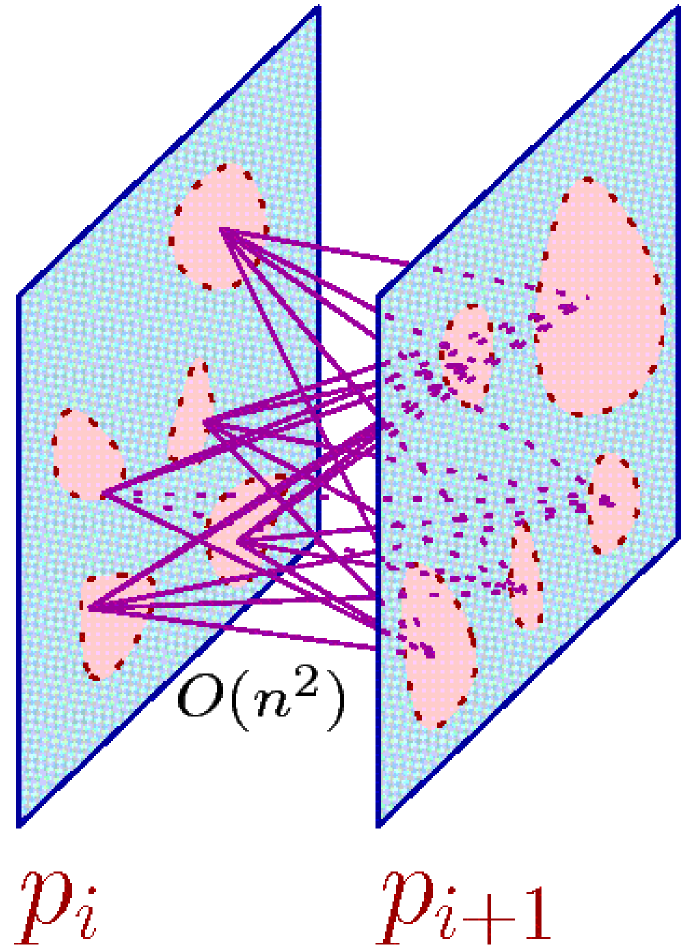
Scenario in Q:



- a_2 is **reachable** from a_1 iff $\delta_F(\overline{p_3 p_4}, \pi(a_1,a_2)) \leq \varepsilon$, where $\pi(a_1,a_2)$ is the shortest path in Q between a_1 and a_2.
- All points in one neighborhood are reachable from all points in another neighborhood, if there exists one reachable pair of points.
- Compute reachability between two neighborhoods in O(n) time.

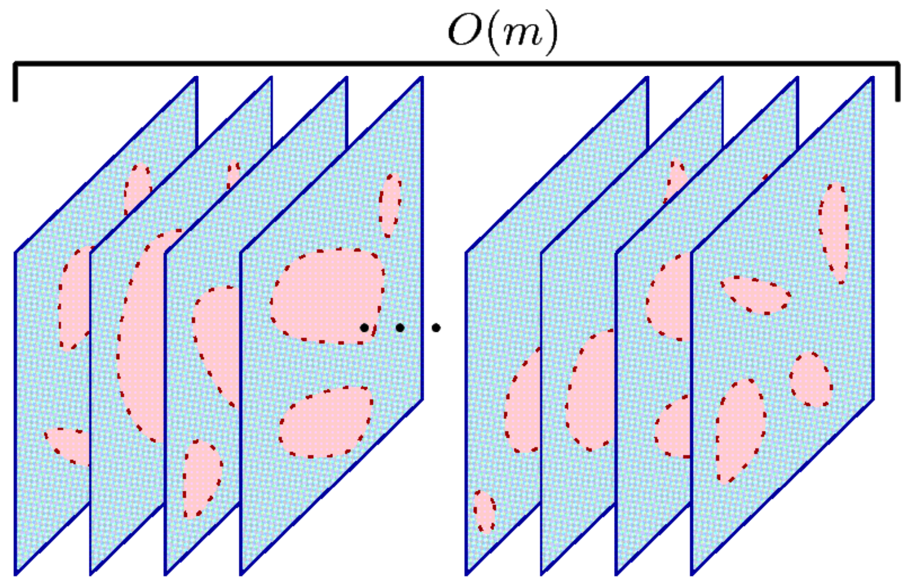
Algorithm: Neighborhoods

- Each slice contains at most O(n) neighborhoods per point.
- There are $O(n^2)$ pairs of neighborhoods to test reachability between, for each pair of slices.



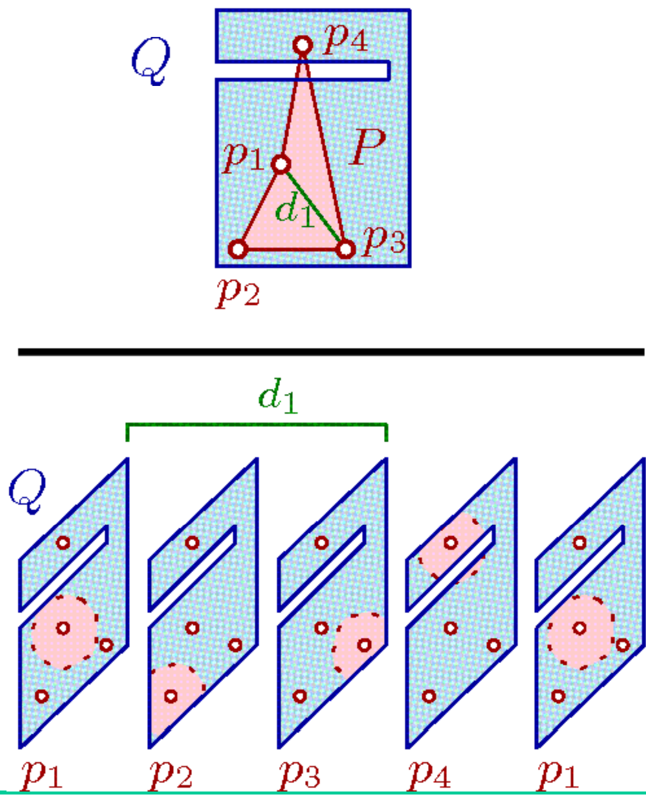
Algorithm: Neighborhoods

- There are O(m) slices, where m=|P|
- We can compute and propagate reachability through free space diagram in $O(n^3 m)$ time
- ⇒ Test whether a reachable path exists, and construct valid set of neighborhoods, in polynomial time.

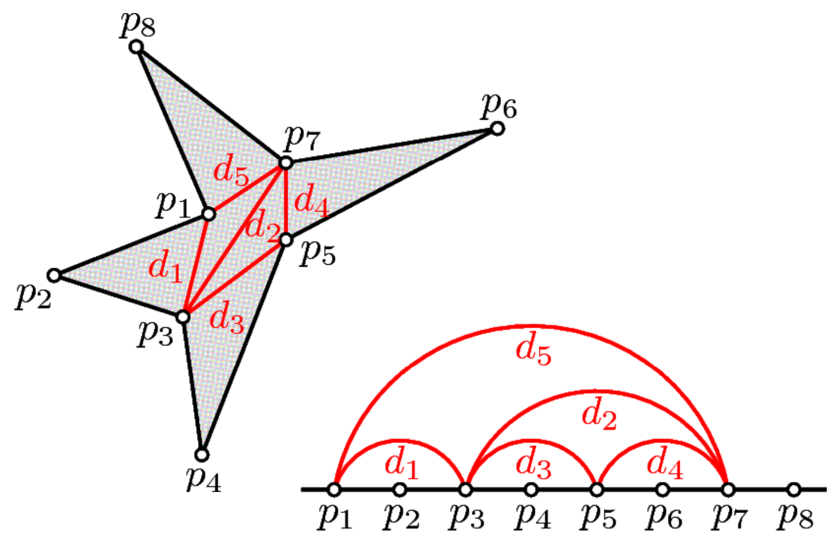


Slight Modification

- So far we have only mapped ∂P, but we have not considered the diagonals of P yet.



- Modify algorithm, by merging slices in a different order:
  - Locally from left to right
  - Merge in **diagonal-nesting-order**



Approach for Partial Fréchet Distance between Simple Polygons

- Since we have to find $R \subseteq Q$, the boundary of R is not known.
  $\Rightarrow$ Simply mapping boundaries does not work.

- We extend simple polygons approach in a different way:

1. Map boundary & check diagonals:

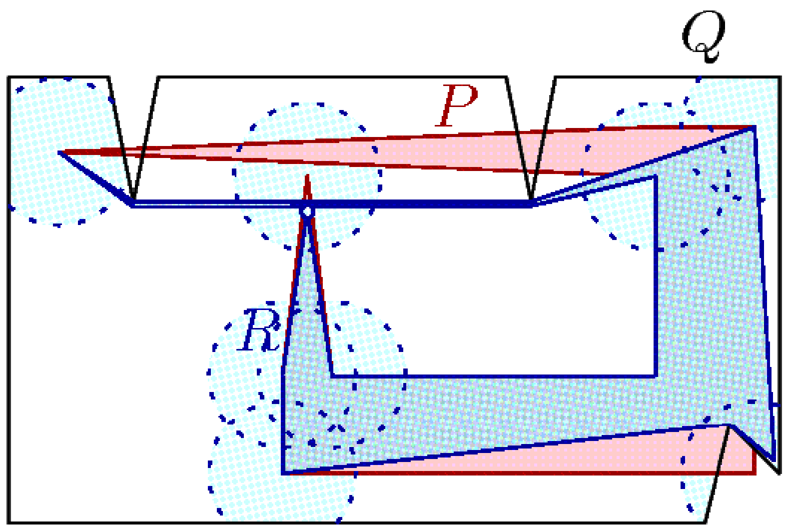
Compute combinatorially equivalent mappings from ∂P to ~~∂Q~~ **some closed curve in Q**, that also ensure small δ_F between diagonals and shortest paths .
$\Rightarrow$ (Q,ε)-valid set of neighborhoods

2. Construct R from (Q,ε)-valid set

Prove that a (Q,ε)-valid set of neighborhoods always contains a valid **simple** polygon $R \subseteq Q$

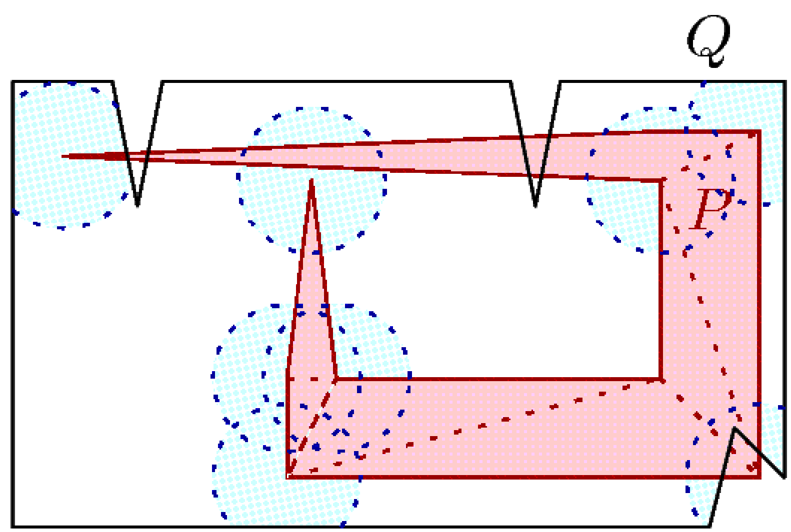
Finding R

- A valid set of neighborhoods
- We want to compute a **simple** polygon R that maps every p_i to a point in its associated neighborhood.



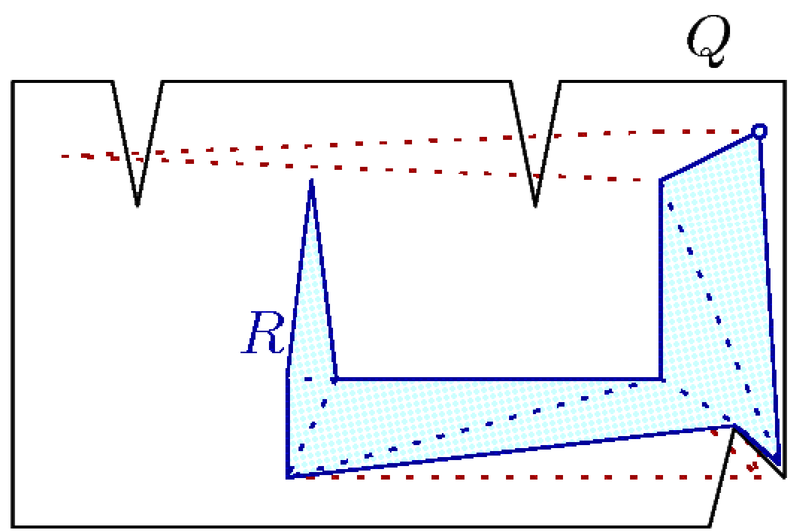
Finding R

- We iteratively construct R by mapping each p_i to a point in its associated neighborhood.
- At each iteration:
  - We show that the points can be mapped to form a simple polygon.
  - We allow remapping points within their neighborhood.
  - By properties of the neighborhoods, δ_F stays $\leq \varepsilon$



Finding R

- Initially, choose one triangle in P.
- Map each point a in its associated neighborhood N_a.
- In each iterative step, add points which are connected to already mapped points.
- If the neighborhood does not contain the original point, map it inside Q and connect previous points with shortest paths.



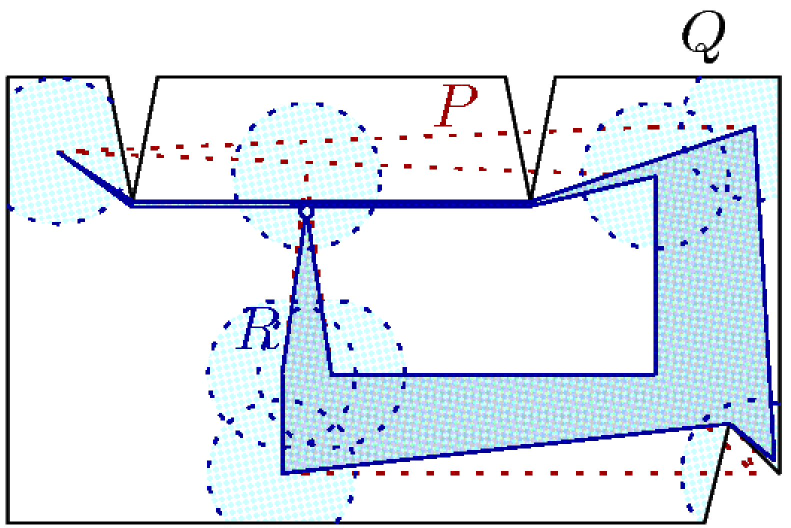
Finding R

- If adding a point and a shortest path yields a self-intersection:
  - The neighborhood around a previous point is crossed.
  - We need to remap to a point below the shortest path.

⇒ In the end we have found a simple polygon R.

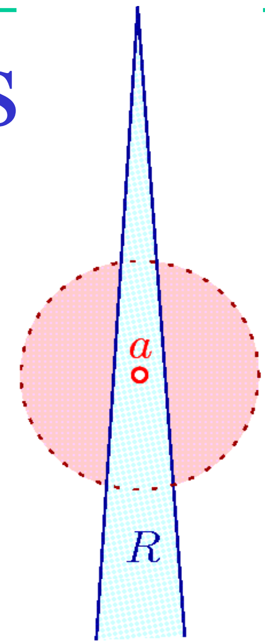
⇒ Vertices of P are mapped to points in associated neighborhoods.

⇒ Hence, $\delta_F(P,R) \leq \varepsilon$.

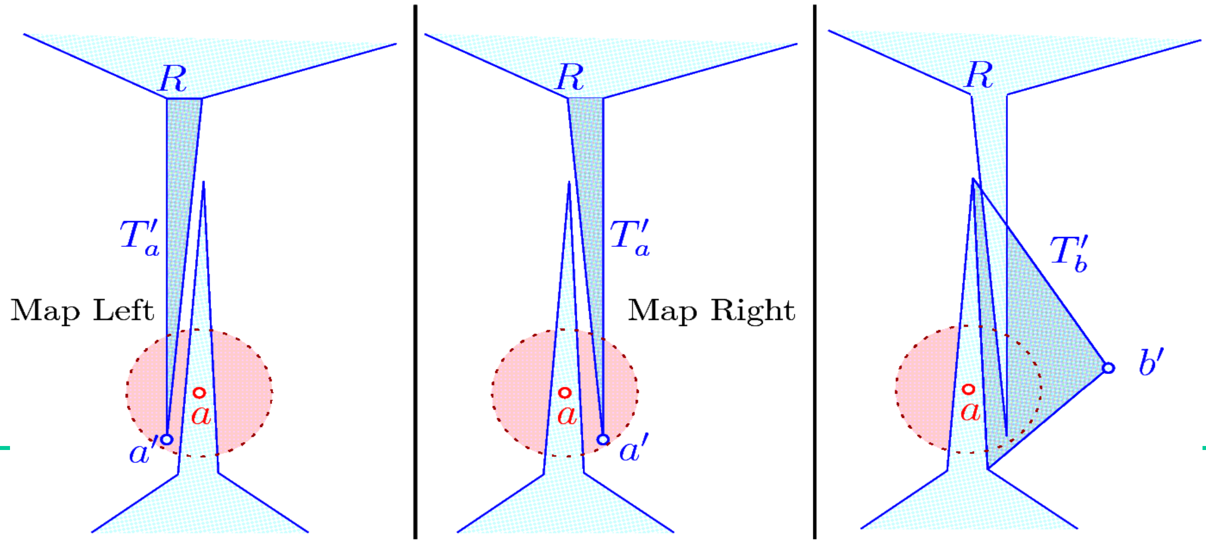


Original Regions

- The neighborhoods of points can be split into multiple disjoint parts by R.
- We must choose one of these two regions to map a to.

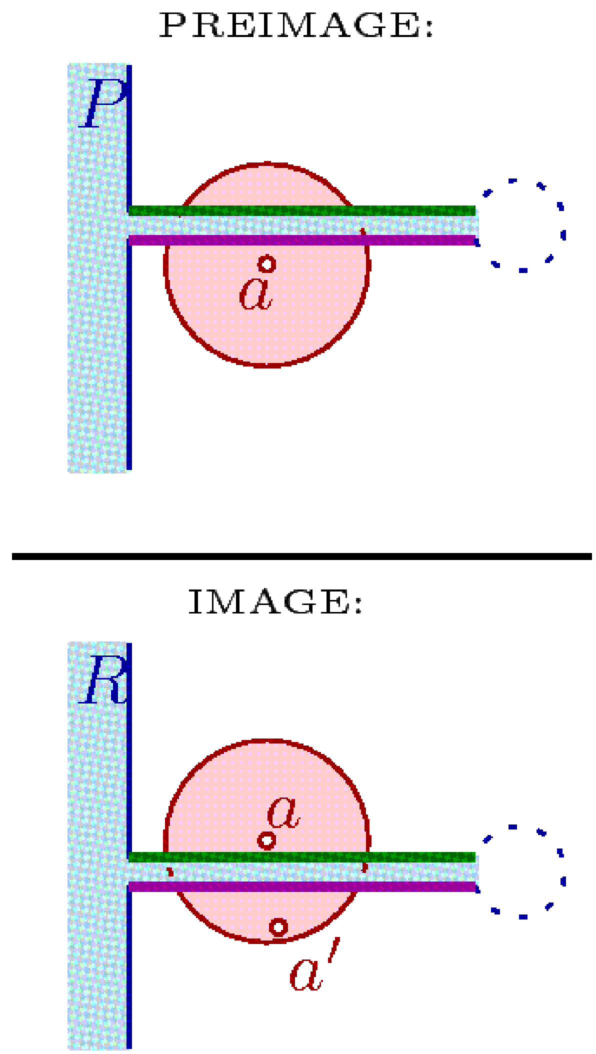


- Unfortunately an arbitrary choice may be invalidated by a later mapping of R.



Original Region

- The key idea is to map a point a' on the same side (relative to R) as a is to the preimage of that portion of R.
  ⇒ Original region



Conclusions:

- We presented the first algorithm for computing partial FD between surfaces. This algorithm runs in polynomial time.
- In the future it would be interesting to consider other variants of partial FD.
- It would also be interesting to consider extending this algorithm other classes of surfaces