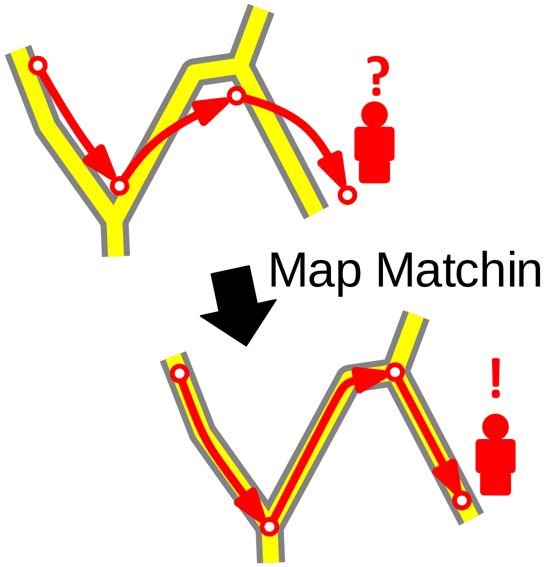
Algorithmen für Geographische Informationssysteme

2. Vorlesung

Alexander Wolff

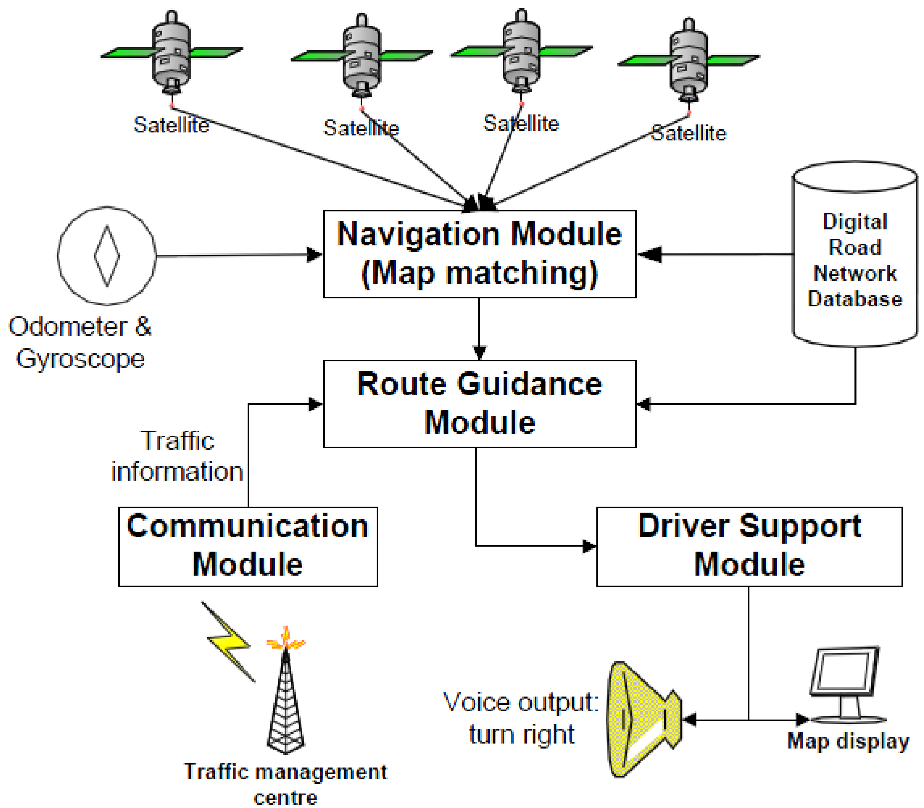
basiert auf Folien von Jan-Henrik Haunert

Map Matching



Map Matching

...als Teil von Fahrzeugnavigationssystemen



Bildquelle: Quddus 2006

Map Matching

Schwierigkeit:

- ungenaue Beobachtungen

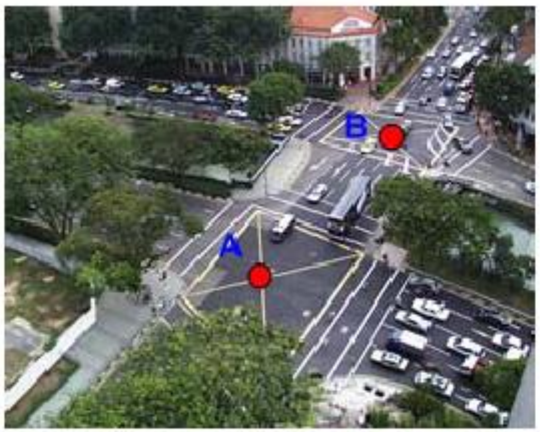
Bildquelle: Wikipedia

GPS: Positionsgenauigkeit ca. 20 m

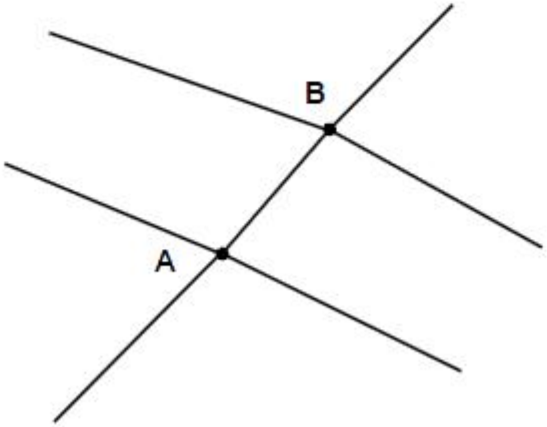
Map Matching

Schwierigkeit:

- ungenaue Beobachtungen
- abstrahierte/vereinfachte Repräsentation des Straßennetzes



tatsächliche Straßenkreuzung



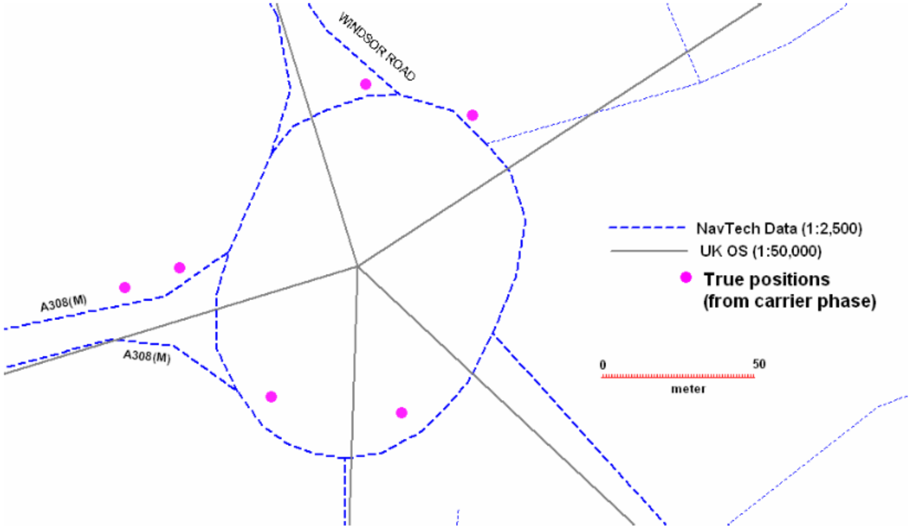
Repräsentation als Graph

Bildquelle: Quddus 2006

Map Matching

Schwierigkeit:

- ungenaue Beobachtungen
- abstrahierte/vereinfachte Repräsentation des Straßennetzes



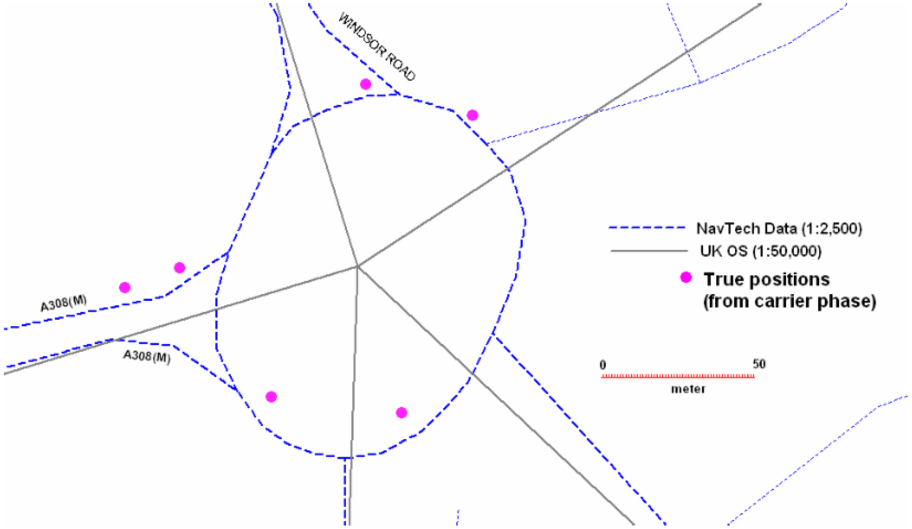
Unterschiede zweier Datensätze

Bildquelle: Quddus 2006

Map Matching

Aufgabenteile:

- Auf welcher Kante befindet sich das Fahrzeug?
- Wo auf der Kante befindet sich das Fahrzeug?

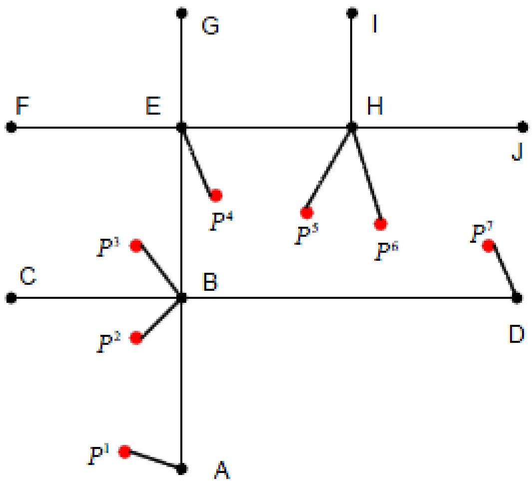


Unterschiede zweier Datensätze

Bildquelle: Quddus 2006

Map Matching

getrennt für jede Positionierung



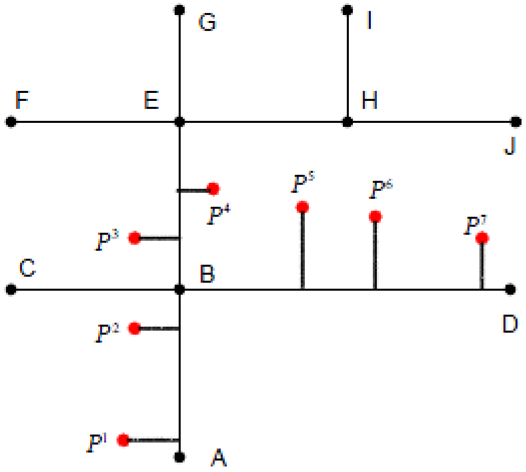
Punkt-zu-Punkt-Zuordnung

Algorithmus?

Bildquelle: Quddus 2006

Map Matching

getrennt für jede Positionierung



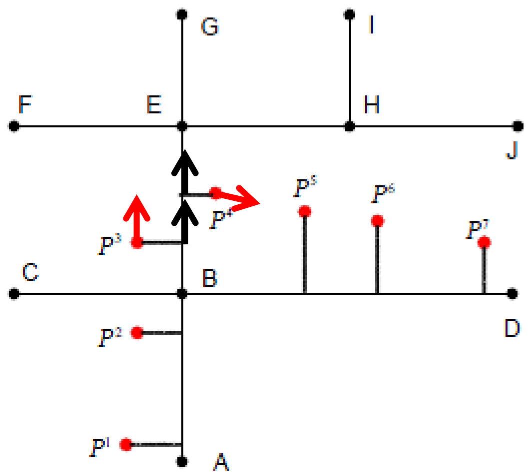
Punkt-zu-Kante-Zuordnung

Bildquelle: Quddus 2006

Map Matching

getrennt für jede Positionierung

weitere Verbesserung:
Vergleiche Fahrtrichtung
mit Kantenrichtung



Punkt-zu-Kante-Zuordnung

Bildquelle: Quddus 2006

Map Matching

Problemformulierung

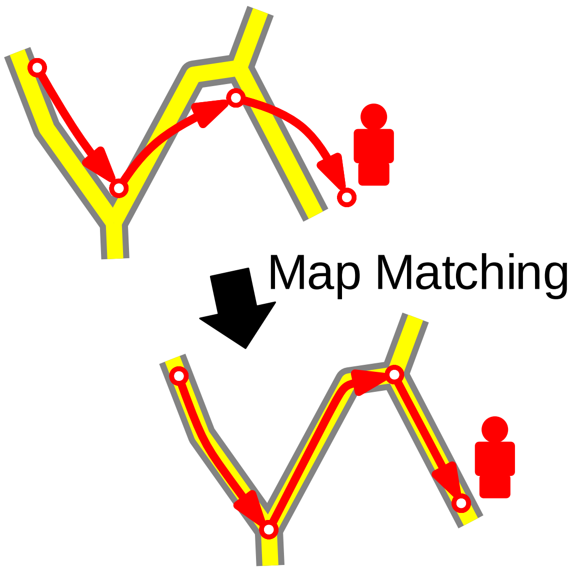
Gegeben:

– Straßennetz als planar eingebetteter Graph $G = (V, E)$

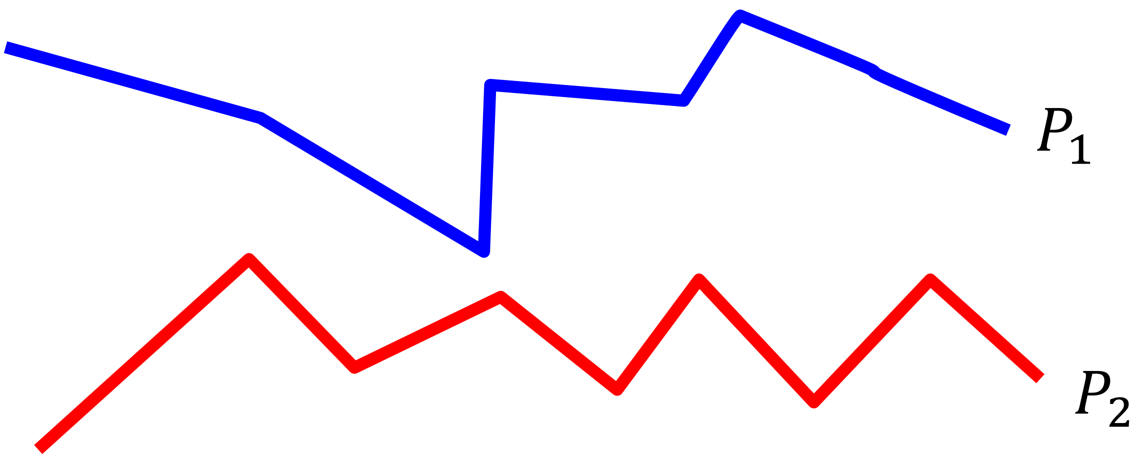
– GPS-Trajektorie als Folge $P = (p_1, p_2, \dots, p_m)$ von Punkten

Gesucht:

Weg in G, der **minimale Distanz** zu P hat.



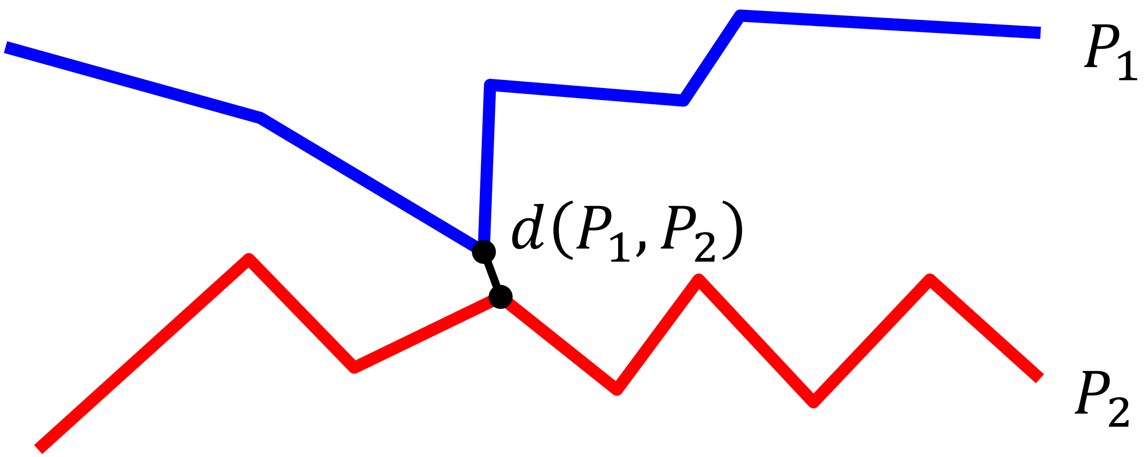
Map Matching



Map Matching

Minimaler euklidischer Abstand zwischen zwei beliebigen Punkten $p_1 \in P_1, p_2 \in P_2$

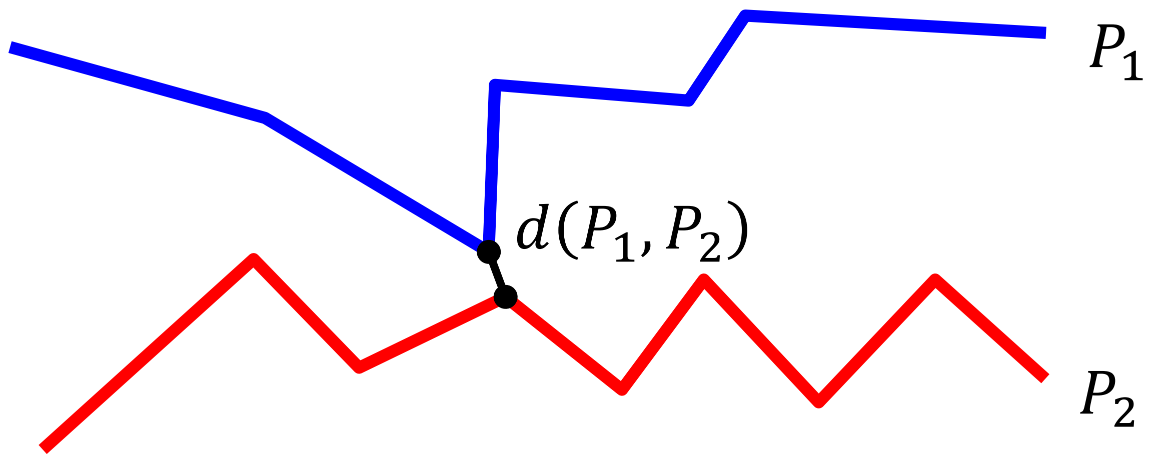
$$d(P_1, P_2) = \min\{d(p_1, p_2) \mid p_1 \in P_1, p_2 \in P_2\}$$



Map Matching

Minimaler euklidischer Abstand zwischen zwei beliebigen Punkten $p_1 \in P_1, p_2 \in P_2$

$$d(P_1, P_2) = \min\{d(p_1, p_2) \mid p_1 \in P_1, p_2 \in P_2\}$$

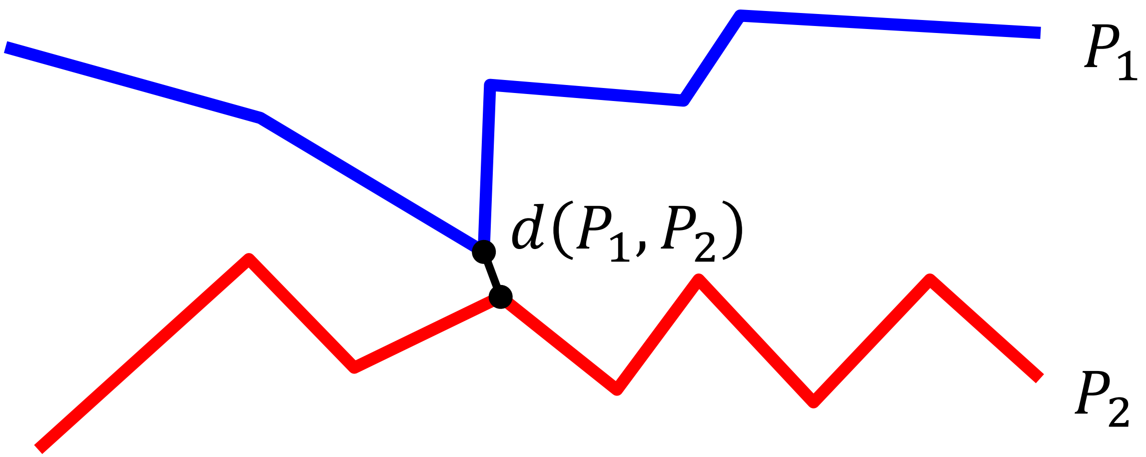


P_1 und P_2 hier: Polygonzüge
allgemeiner: Punktmengen

Map Matching

Minimaler euklidischer Abstand zwischen zwei beliebigen Punkten $p_1 \in P_1, p_2 \in P_2$

$$d(P_1, P_2) = \min\{d(p_1, p_2) \mid p_1 \in P_1, p_2 \in P_2\}$$



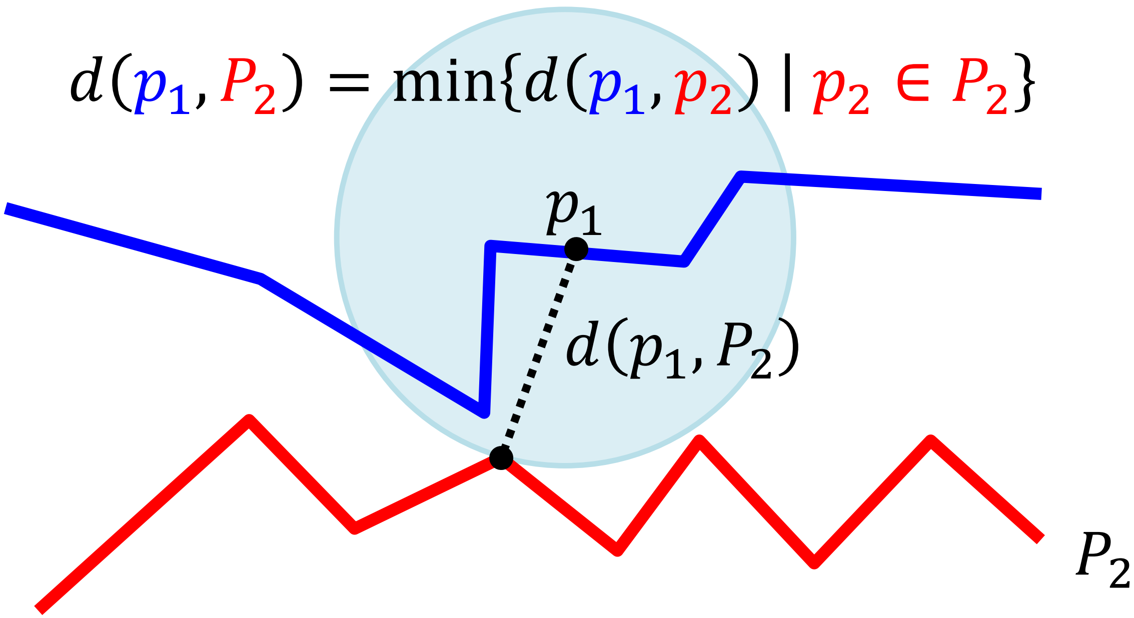
Wird zu Null, wenn sich P_1 und P_2 schneiden!

Als Distanzmaß ungeeignet!

Map Matching

Minimaler euklidischer Abstand zwischen
einem festen Punkt $p_1 \in P_1$ und einem beliebigen Punkt $p_2 \in P_2$

$$d(p_1, P_2) = \min\{d(p_1, p_2) \mid p_2 \in P_2\}$$

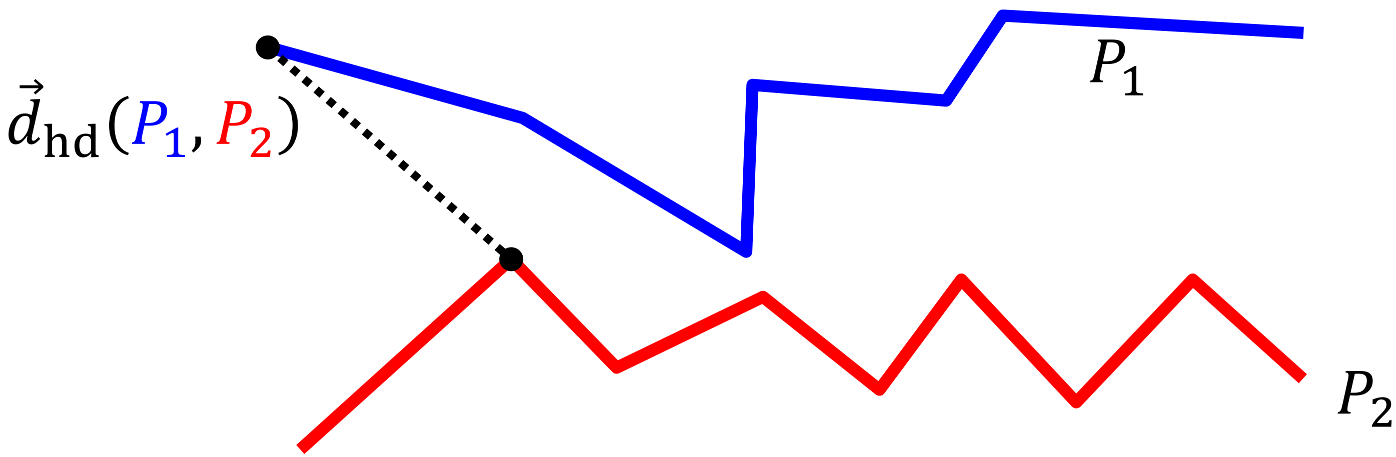


Map Matching

gerichtete Hausdorff-Distanz

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$

mit $$d(p_1, P_2) = \min\{d(p_1, p_2) \mid p_2 \in P_2\}$$

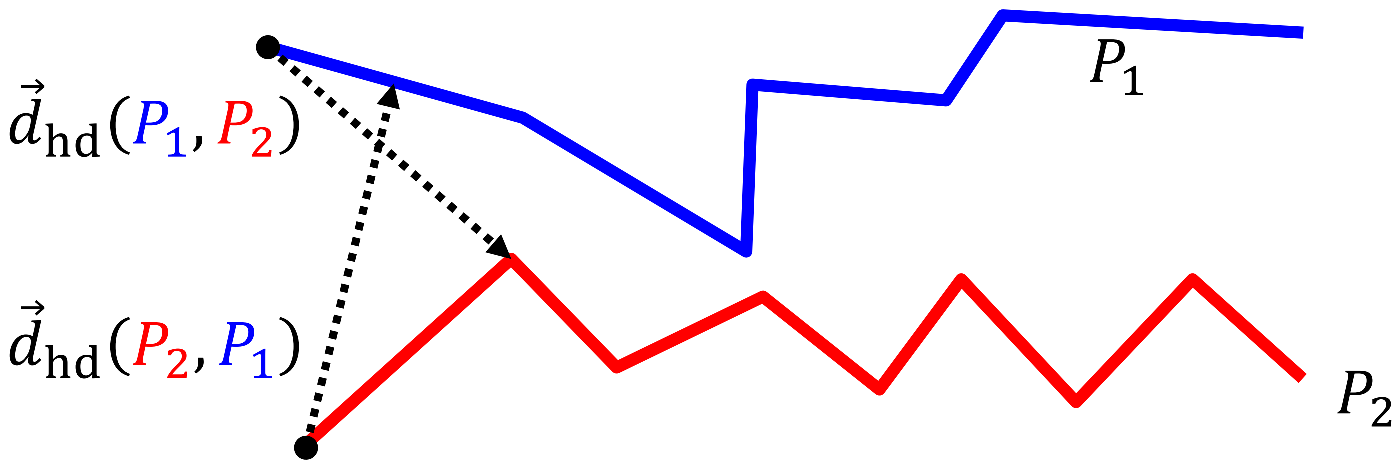


Map Matching

gerichtete Hausdorff-Distanz

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$

mit $$d(p_1, P_2) = \min\{d(p_1, p_2) \mid p_2 \in P_2\}$$



Achtung: Allgemein gilt $\vec{d}_{\mathrm{hd}}(P_1, P_2) \neq \vec{d}_{\mathrm{hd}}(P_2, P_1)$.

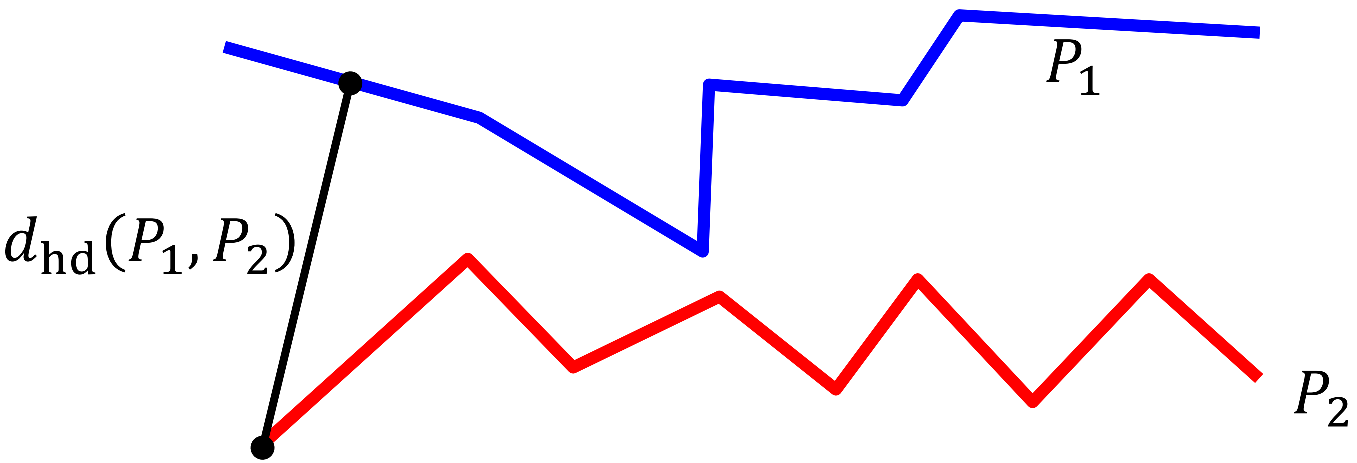
Map Matching

Hausdorff-Distanz

$$d_{\mathrm{hd}}(P_1, P_2) = \max\left\{\vec{d}_{\mathrm{hd}}(P_1, P_2), \vec{d}_{\mathrm{hd}}(P_2, P_1)\right\}$$

mit $\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$

und $d(p_1, P_2) = \min\{d(p_1, p_2) \mid p_2 \in P_2\}$

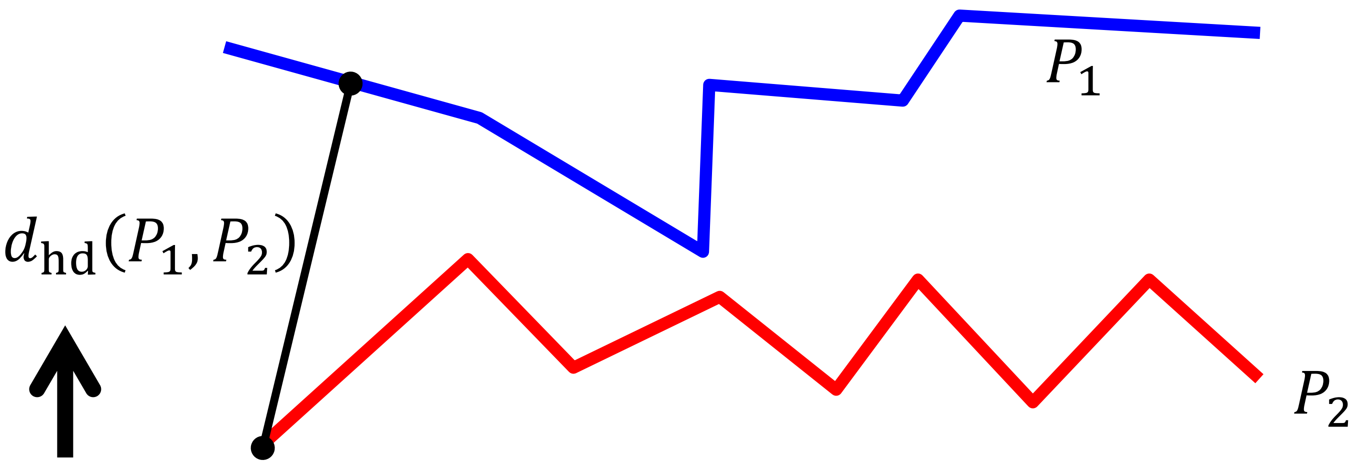


Algorithmus?

Map Matching

Hausdorff-Distanz

Berechnung für zwei Polygonzüge:

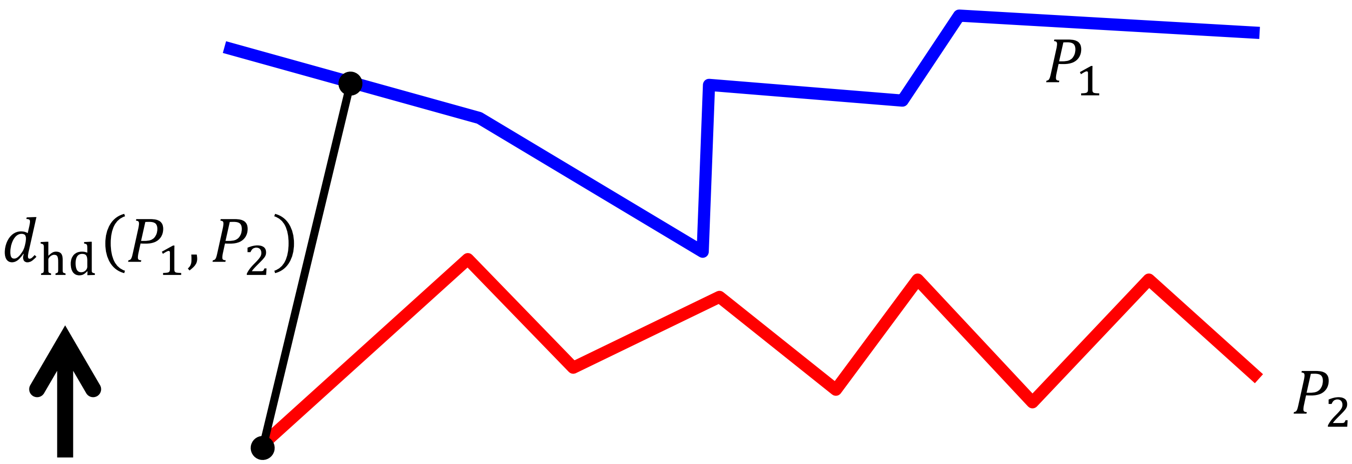


Bezieht immer eine **Ecke & eine Kante**
oder **zwei Ecken** ein.

Map Matching

Hausdorff-Distanz

Berechnung für zwei Polygonzüge:



Bezieht immer eine **Ecke & eine Kante** ***oder*** **zwei Ecken** ein.

Naiver Algorithmus braucht $O(mn)$ Zeit.

Map Matching

Hausdorff-Distanz

Besserer Algorithmus:

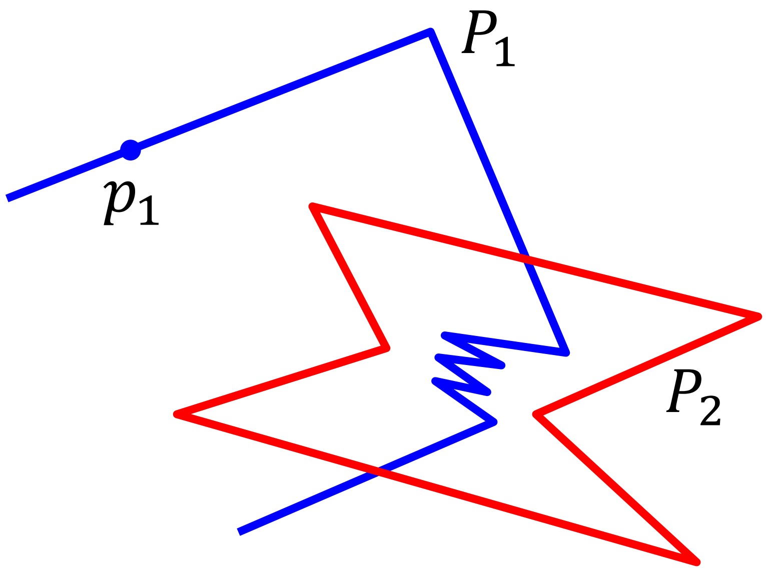
Alt, Behrends, Blömer, 1995: Approximate Matching of Polygonal Shapes. *Annals of Mathematics and Artificial Intelligence*, 13(1995), 251–265.

Map Matching

Hausdorff-Distanz

Besserer Algorithmus:

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$



Map Matching

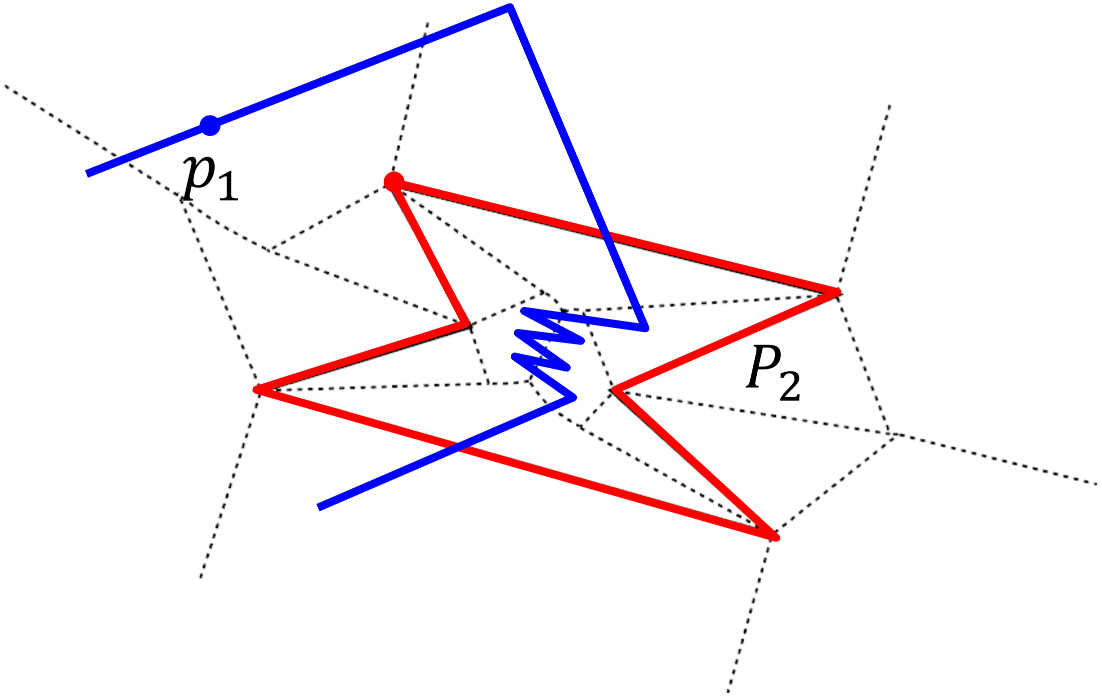
Hausdorff-Distanz

Besserer Algorithmus:

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$

Voronoi-Diagramm für P_2:

- kann in $O(n \log n)$ Zeit berechnet werden und
- hat $O(n)$ Kanten.

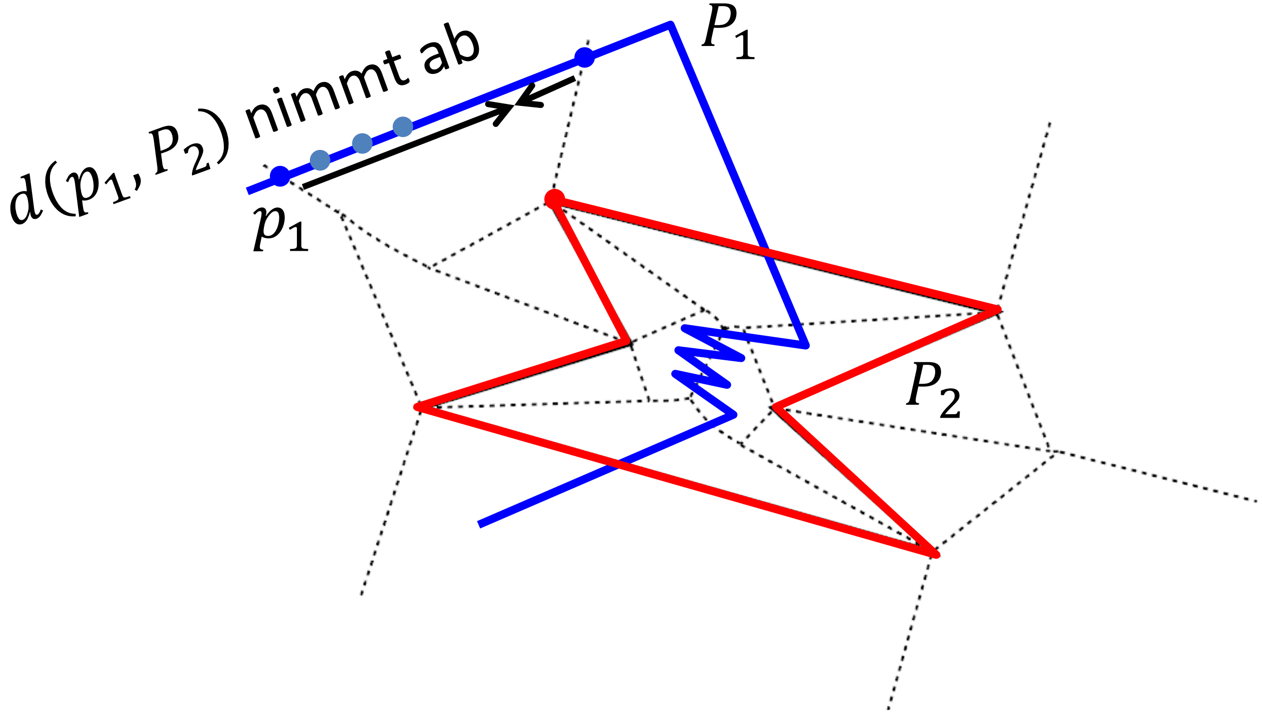


Map Matching

Hausdorff-Distanz

Besserer Algorithmus:

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$

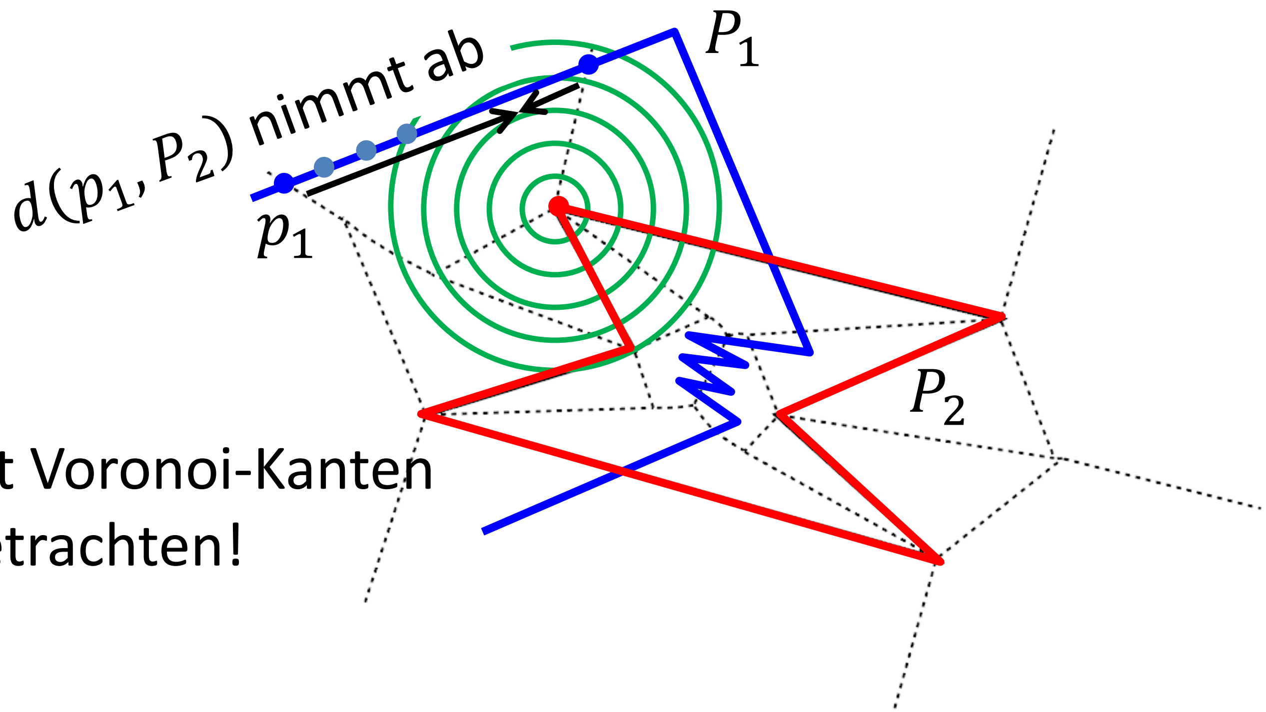


Map Matching

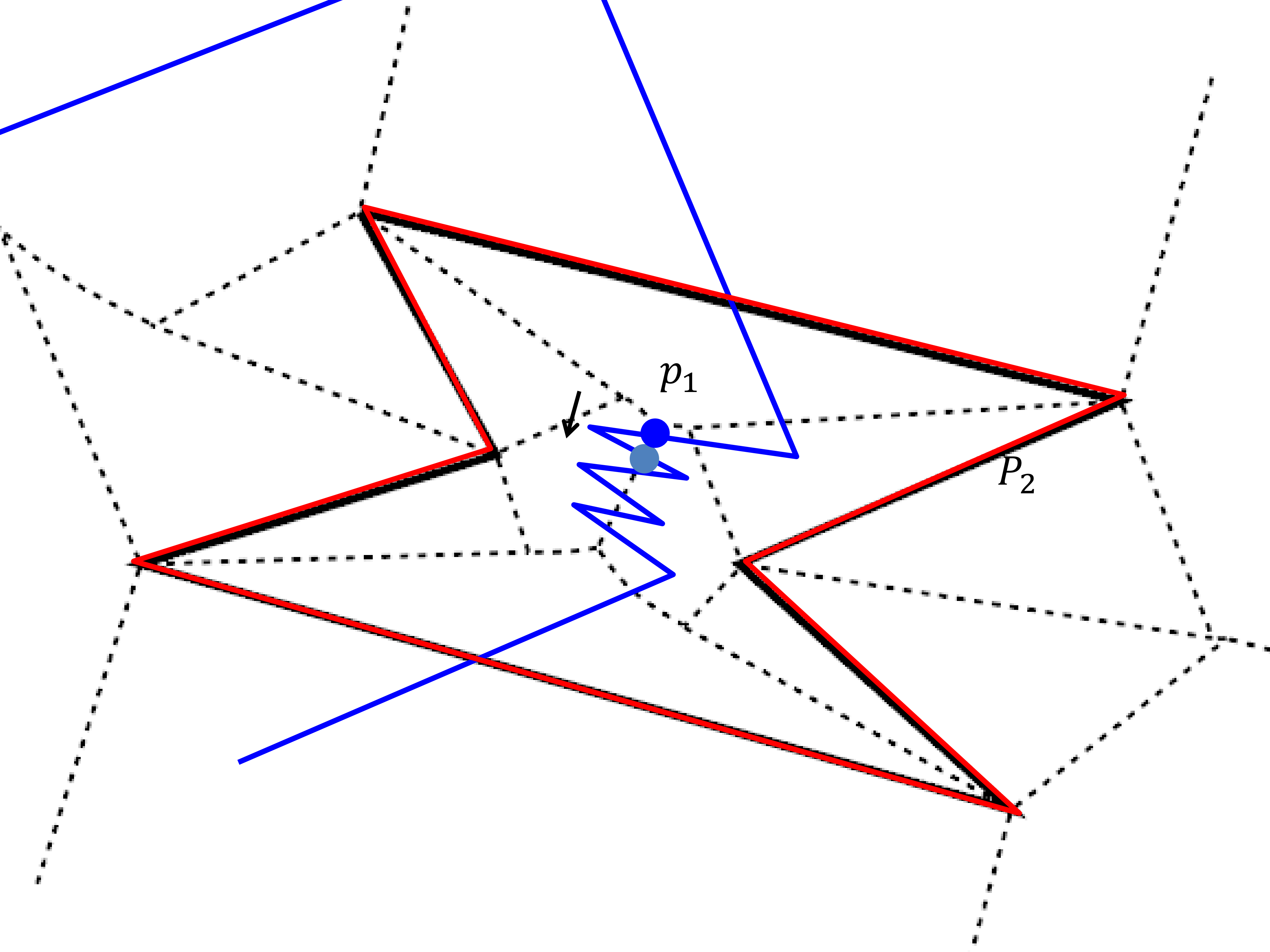
Hausdorff-Distanz

Besserer Algorithmus:

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$



Für p_1 reicht es aus,
– Schnitte von P_1 mit Voronoi-Kanten
– Ecken von P_1 zu betrachten!



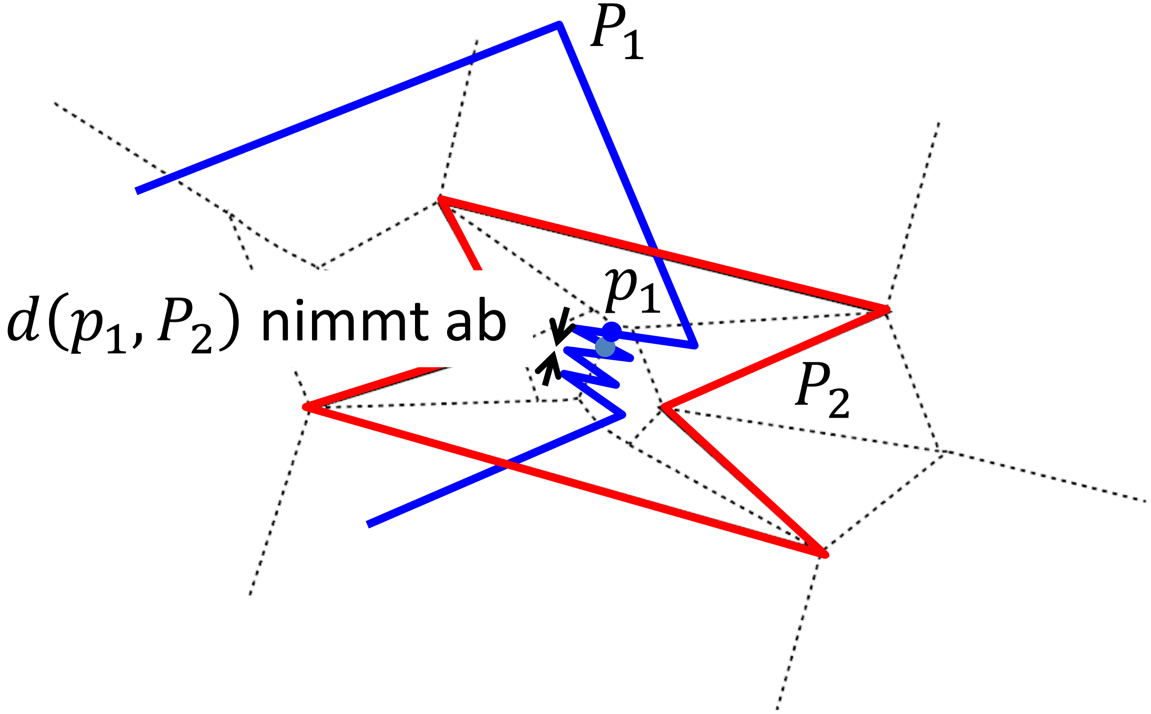
p_1
P_2

Map Matching

Hausdorff-Distanz

Besserer Algorithmus:

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$



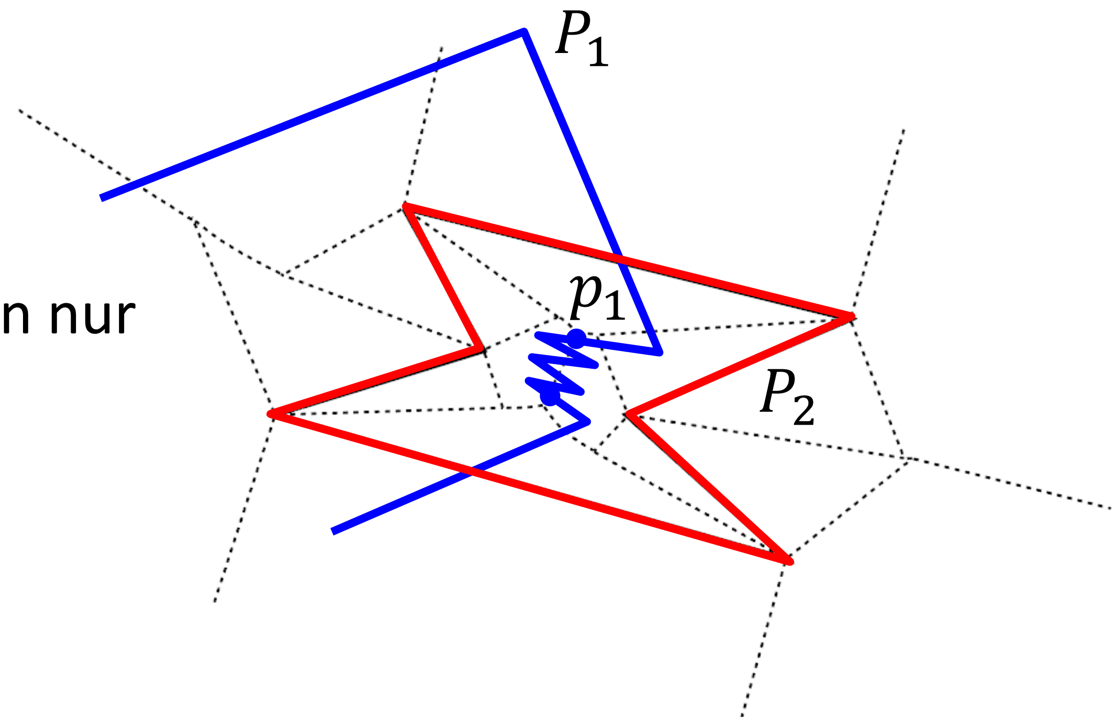
Map Matching

Hausdorff-Distanz

Besserer Algorithmus:

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$

Pro Voronoi-Kante kommen nur zwei Schnitte für p_1 in Betracht!



Map Matching

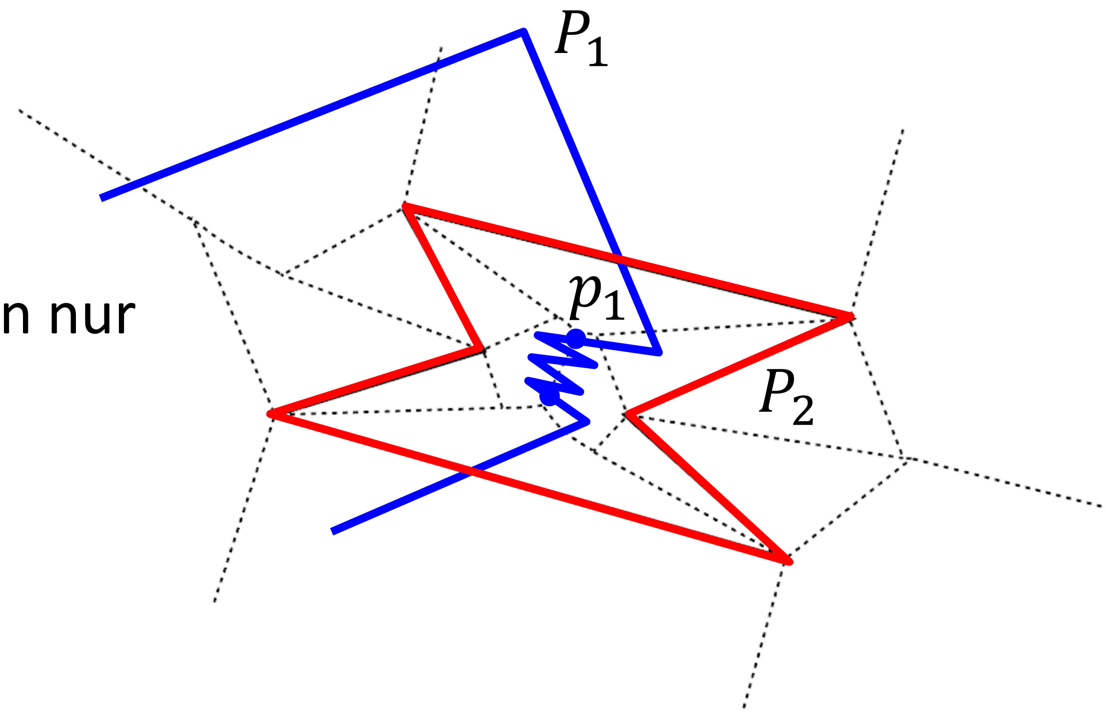
Hausdorff-Distanz

diskrete Menge mit $O(m+n)$ Elementen

Besserer Algorithmus:

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in S\}$$

Pro Voronoi-Kante kommen nur zwei Schnitte für p_1 in Betracht!



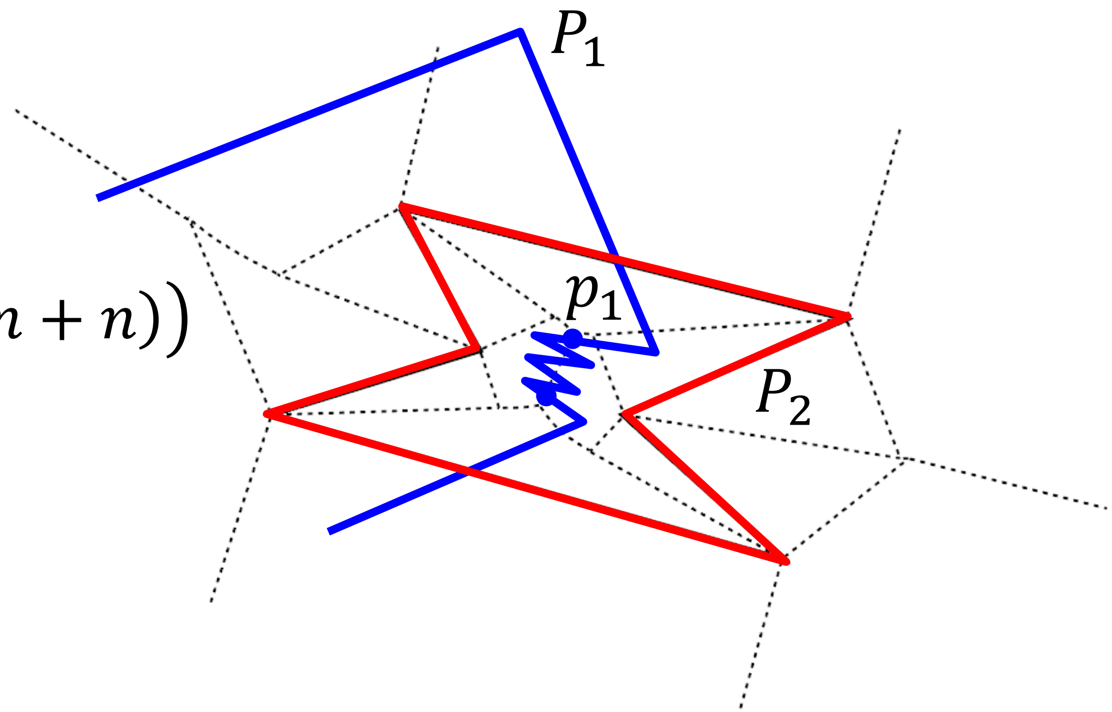
Map Matching

Hausdorff-Distanz

diskrete Menge mit $O(m+n)$ Elementen

Besserer Algorithmus:

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in S\}$$



S kann in $O\big((m+n)\log(m+n)\big)$ gefunden werden.
(Sweepline-Algorithmus)

Map Matching

Hausdorff-Distanz

Besserer Algorithmus:

Laufzeit:

Voronoi-Diagramm	$O(n \log n)$
Menge S	$O\big((m+n)\log(m+n)\big)$
Maximum finden	$O(m+n)$

Map Matching

Hausdorff-Distanz

Besserer Algorithmus:

Laufzeit:

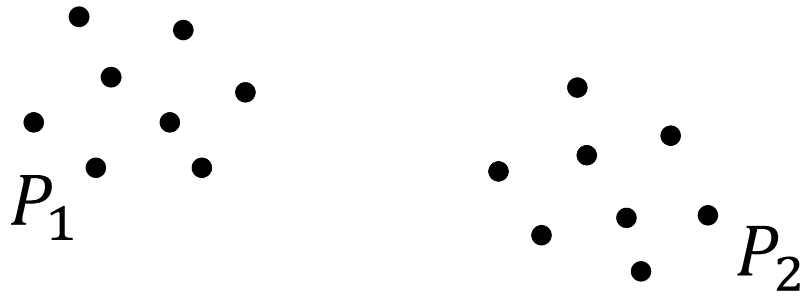
Voronoi-Diagramm	$O(n \log n)$
Menge S	$O\big((m+n)\log(m+n)\big)$
Maximum finden	$O(m+n)$
Gesamt	$\boldsymbol{O\big((m+n)\log(m+n)\big)}$

Map Matching

Hausdorff-Distanz

Typische Aufgabe (Point-Pattern-Matching)

- Gegeben zwei diskrete Punktmengen P_1, P_2
- Finde Rotation/Verschiebung f,
  so dass $d_{\mathrm{hd}}(f(P_1), P_2)$ minimal ist.

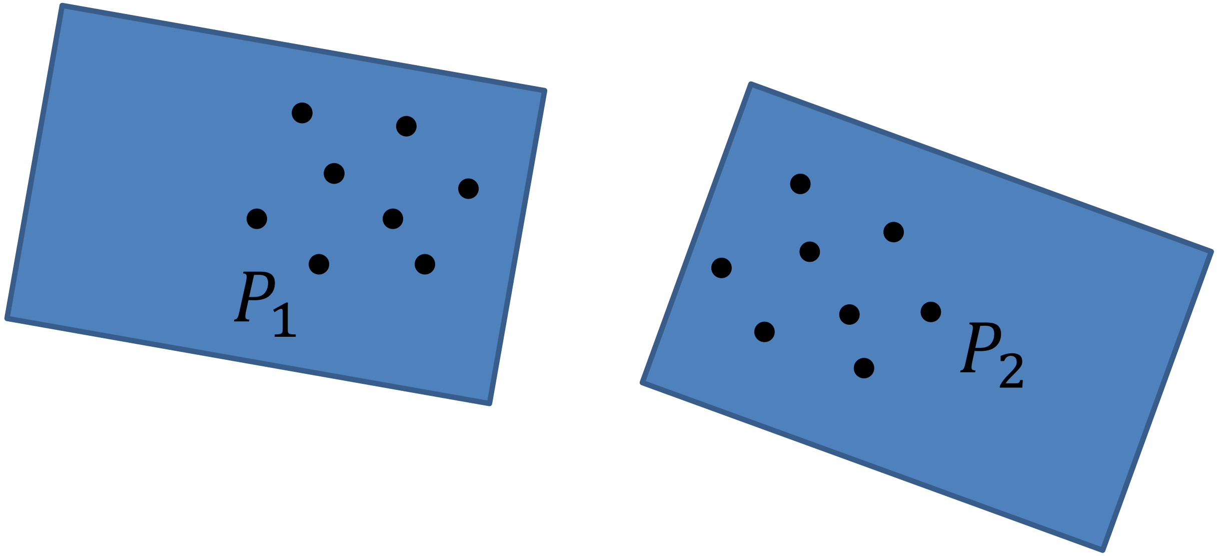


Map Matching

Hausdorff-Distanz

Typische Aufgabe (Point-Pattern-Matching)

- Gegeben zwei diskrete Punktmengen P_1, P_2
- Finde Rotation/Verschiebung f,
  so dass $d_{\mathrm{hd}}(f(P_1), P_2)$ minimal ist.

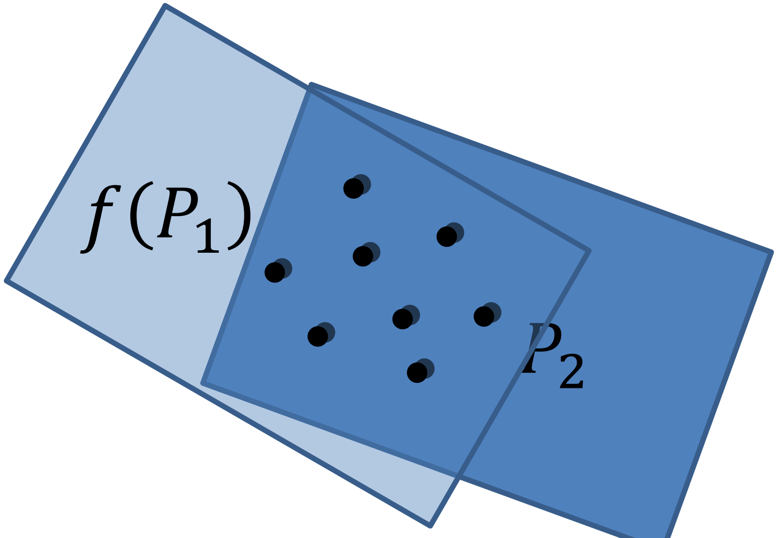


Map Matching

Hausdorff-Distanz

Typische Aufgabe (Point-Pattern-Matching)

- Gegeben zwei diskrete Punktmengen P_1, P_2
- Finde Rotation/Verschiebung f, so dass $d_{\mathrm{hd}}(f(P_1), P_2)$ minimal ist.

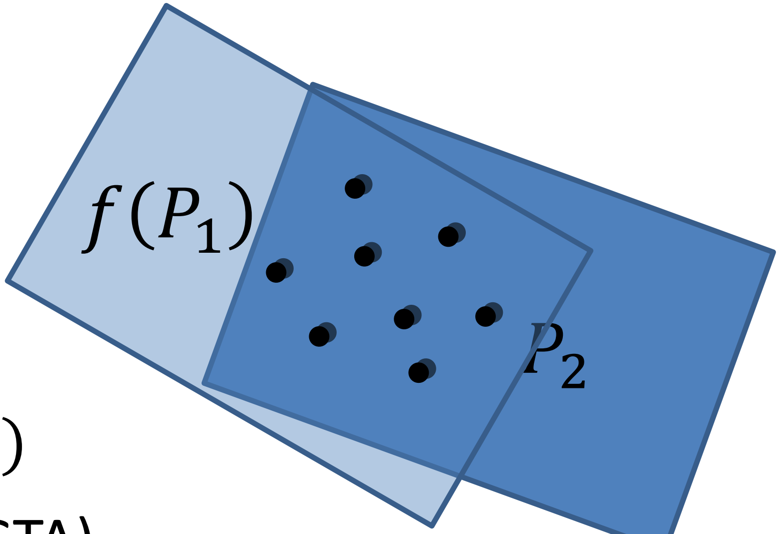


Map Matching

Hausdorff-Distanz

Typische Aufgabe (Point-Pattern-Matching)

- Gegeben zwei diskrete Punktmengen P_1, P_2
- Finde Rotation/Verschiebung f,
  so dass $d_{\mathrm{hd}}(f(P_1), P_2)$ minimal ist.



Lösbar in
$O(|P_1|^3 |P_2|^3 \log^2 |P_1||P_2|)$
Zeit (Chew et al. 1997, CGTA)

Map Matching

gerichtete Hausdorff-Distanz

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$

mit $$d(p_1, P_2) = \min\{d(p_1, p_2) \mid p_2 \in P_2\}$$

Stellt für $p_1 \in P_1$ einen Bezug zu einem Punkt $p_2 \in P_2$ her.

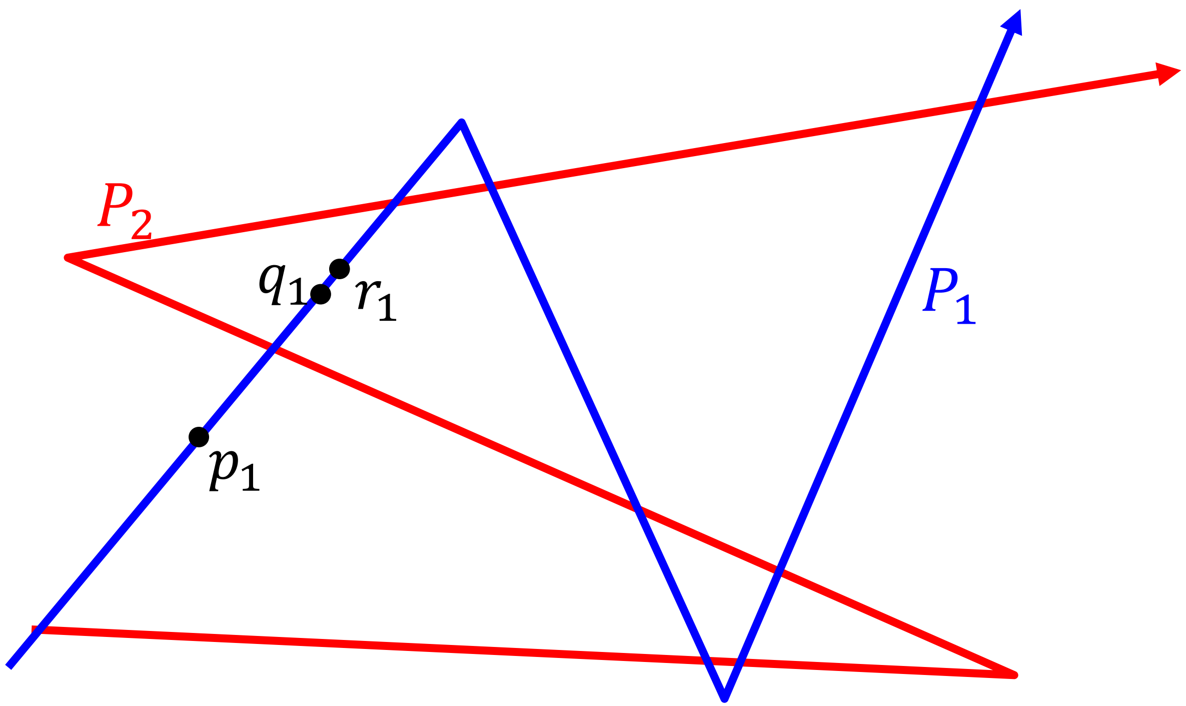
Für Map Matching wichtig!

Map Matching

gerichtete Hausdorff-Distanz

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$

mit $$d(p_1, P_2) = \min\{d(p_1, p_2) \mid p_2 \in P_2\}$$

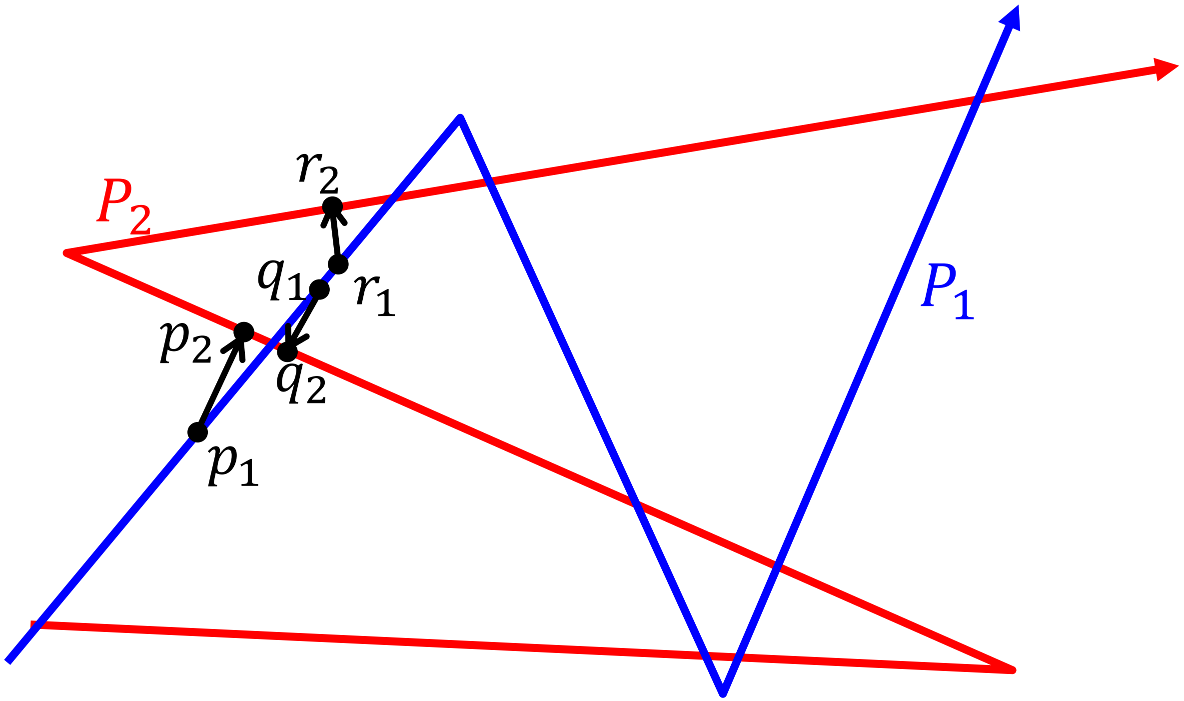


Map Matching

gerichtete Hausdorff-Distanz

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$

mit $$d(p_1, P_2) = \min\{d(p_1, p_2) \mid p_2 \in P_2\}$$

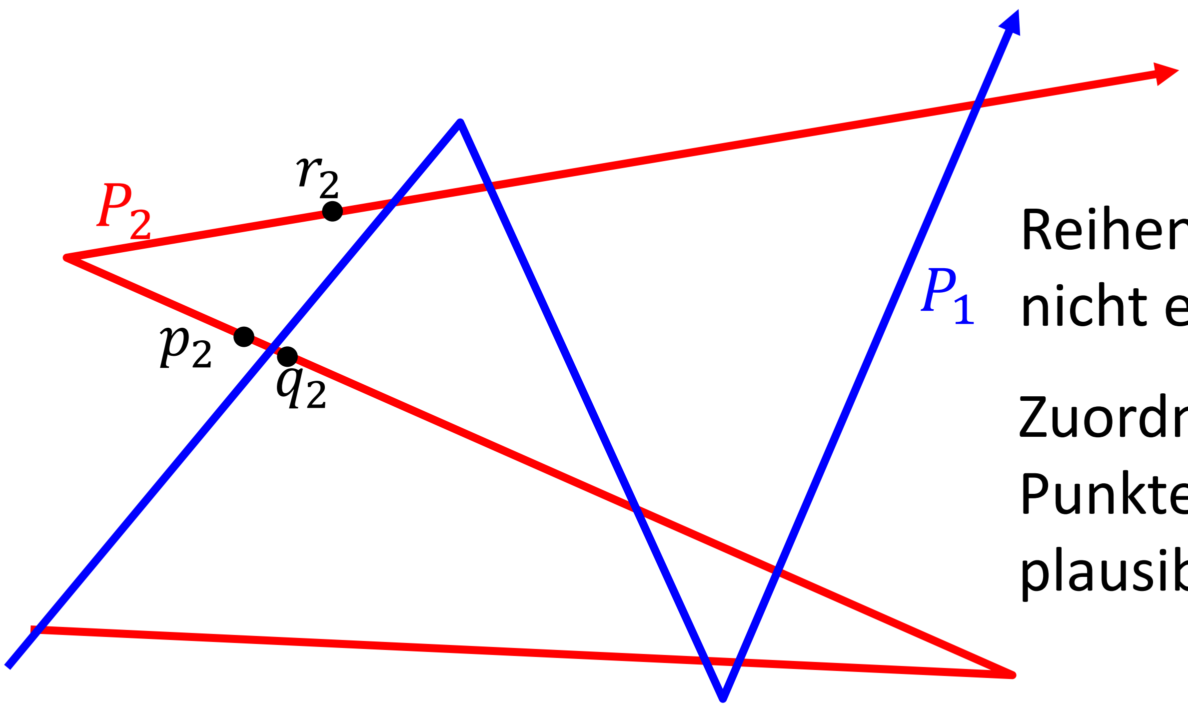


Map Matching

gerichtete Hausdorff-Distanz

$$\vec{d}_{\mathrm{hd}}(P_1, P_2) = \max\{d(p_1, P_2) \mid p_1 \in P_1\}$$

mit $$d(p_1, P_2) = \min\{d(p_1, p_2) \mid p_2 \in P_2\}$$



Reihenfolge pqr wird nicht eingehalten!

Zuordnung der Punkte nicht plausibel!

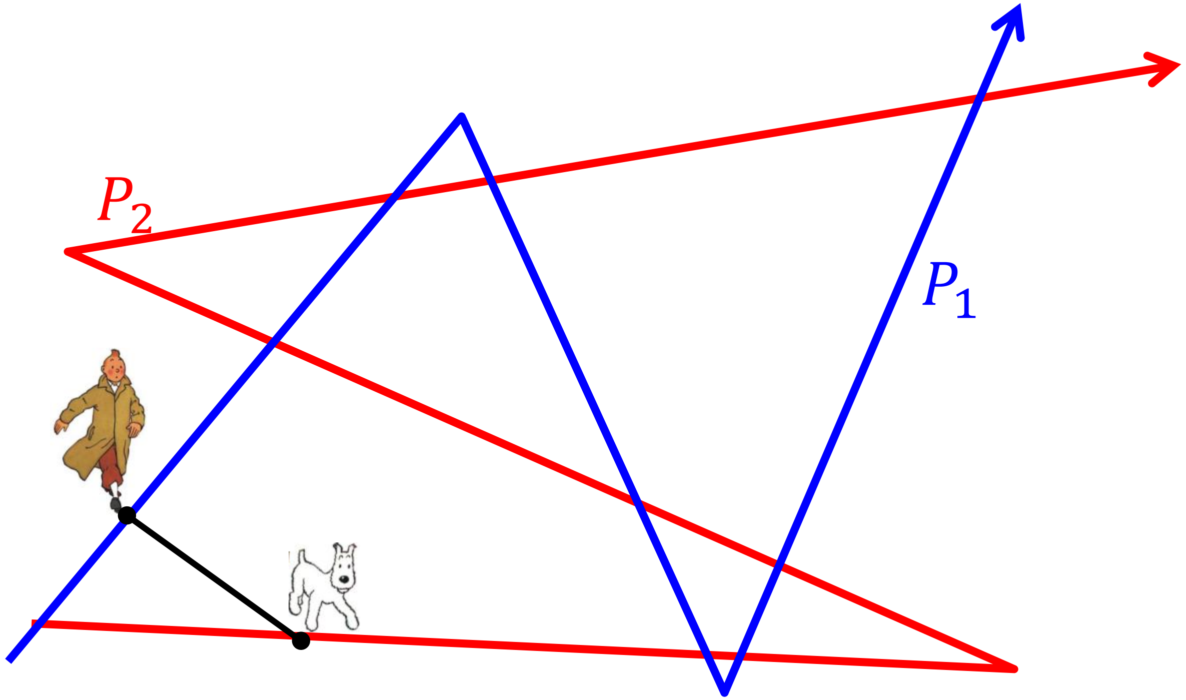
Fréchet-Distanz

Herrchen läuft entlang P_1 ohne jemals umzukehren.

Hund läuft entlang P_2 ohne jemals umzukehren.

Herrchen und Hund sind über **Hundeleine** verbunden.

Wie lang ist die Hundeleine mindestens?



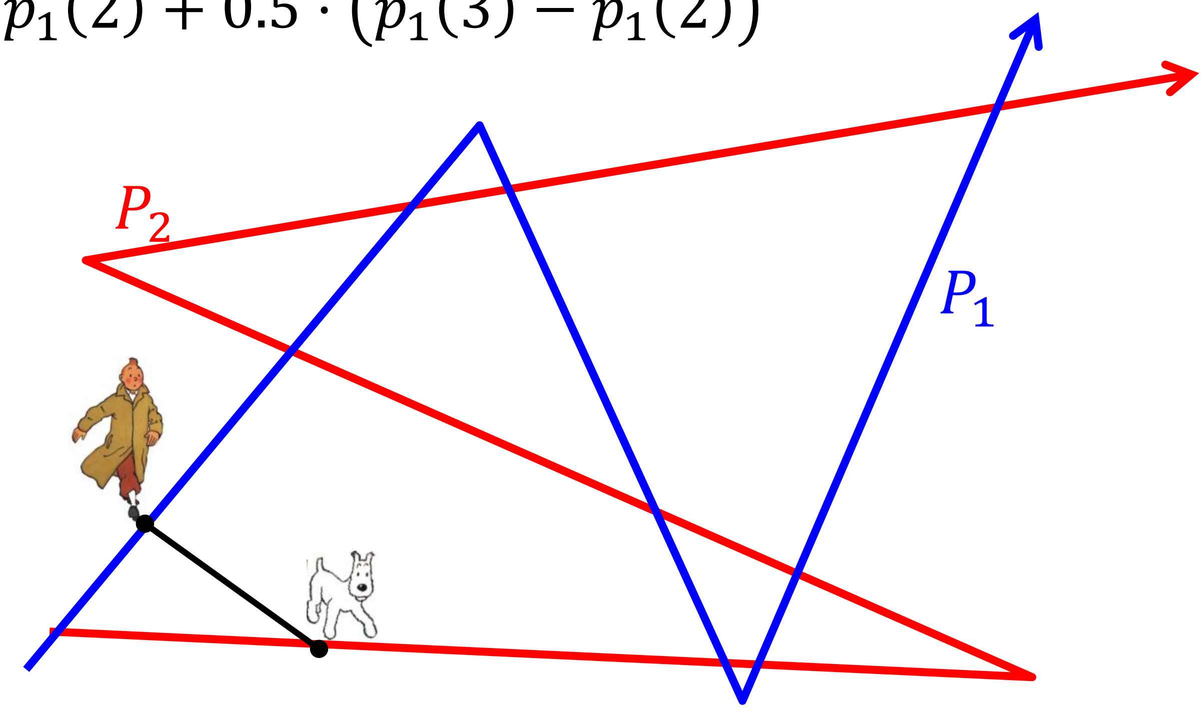
Fréchet-Distanz

Polygonzug als parametrisierte Kurve:

$p_1(t)$ mit $0 \leq t \leq m-1$ ist ein Punkt auf P_1.
Für ganzzahlige t ist $p_1(t)$ die $(t+1)$-te Ecke von P_1.
Ansonsten ist $p_1(t) = p_1(\lfloor t \rfloor) + (t - \lfloor t \rfloor) \cdot \left(p_1(\lfloor t \rfloor + 1) - p_1(\lfloor t \rfloor) \right)$

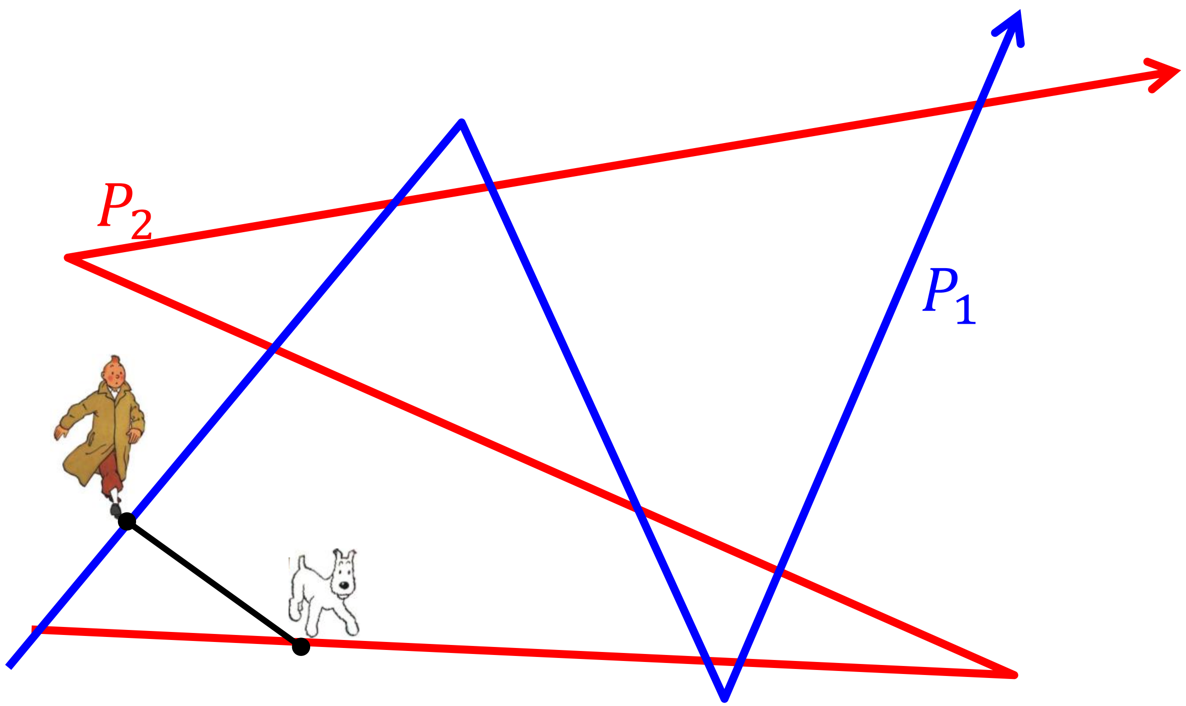
Beispiel:

$p_1(2.5) = p_1(2) + 0.5 \cdot \left(p_1(3) - p_1(2) \right)$



Fréchet-Distanz

Definition: $d_{\text{fréchet}} = \qquad \max\limits_{t\in[0,1]} \left\{ d\left(p_1(\alpha(t)), p_2(\beta(t))\right)\right\}$



Fréchet-Distanz

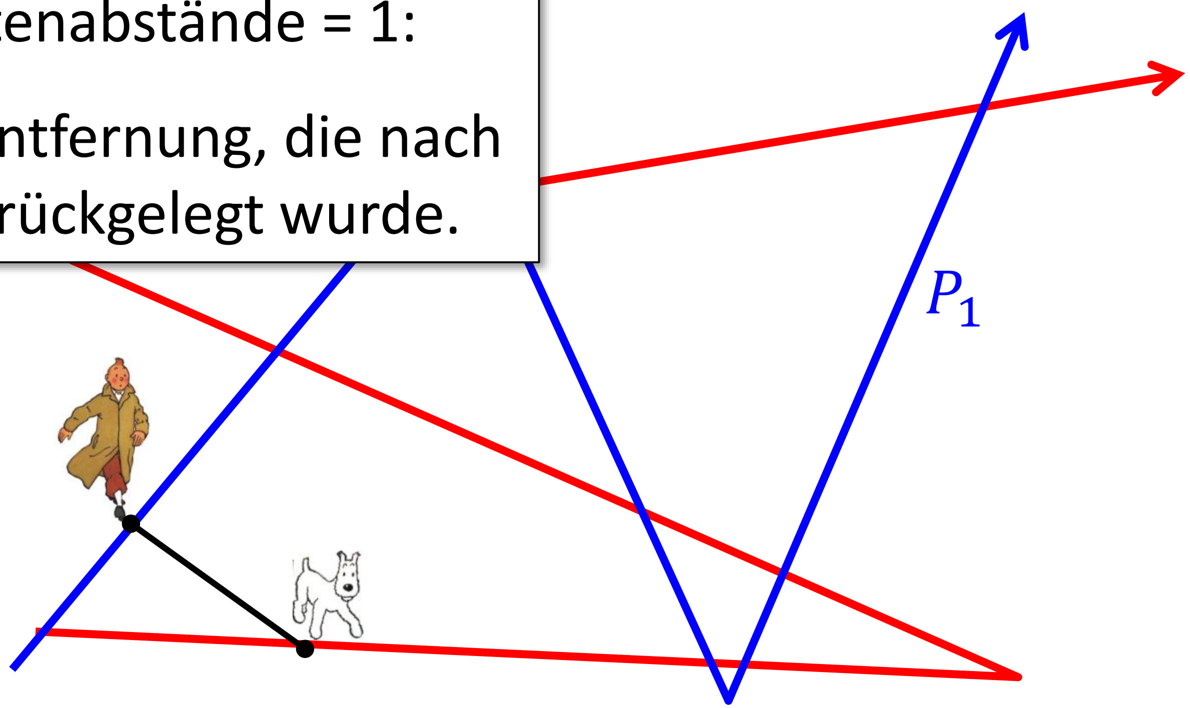
Definition:

$$d_{\text{fréchet}} = \min_{\substack{\alpha:[0,1]\to[0,m-1] \\ \beta:[0,1]\to[0,n-1]}} \max_{t\in[0,1]} \left\{ d\left(p_1(\alpha(t)), p_2(\beta(t)) \right) \right\}$$

wobei α und β **monoton wachsende** und **stetige Funktionen** sind und $\alpha(0) = \beta(0) = 0, \alpha(1) = m - 1, \beta(1) = n - 1.$

Für Knotenabstände = 1:

$\alpha(t)$ = Entfernung, die nach Zeit t zurückgelegt wurde.

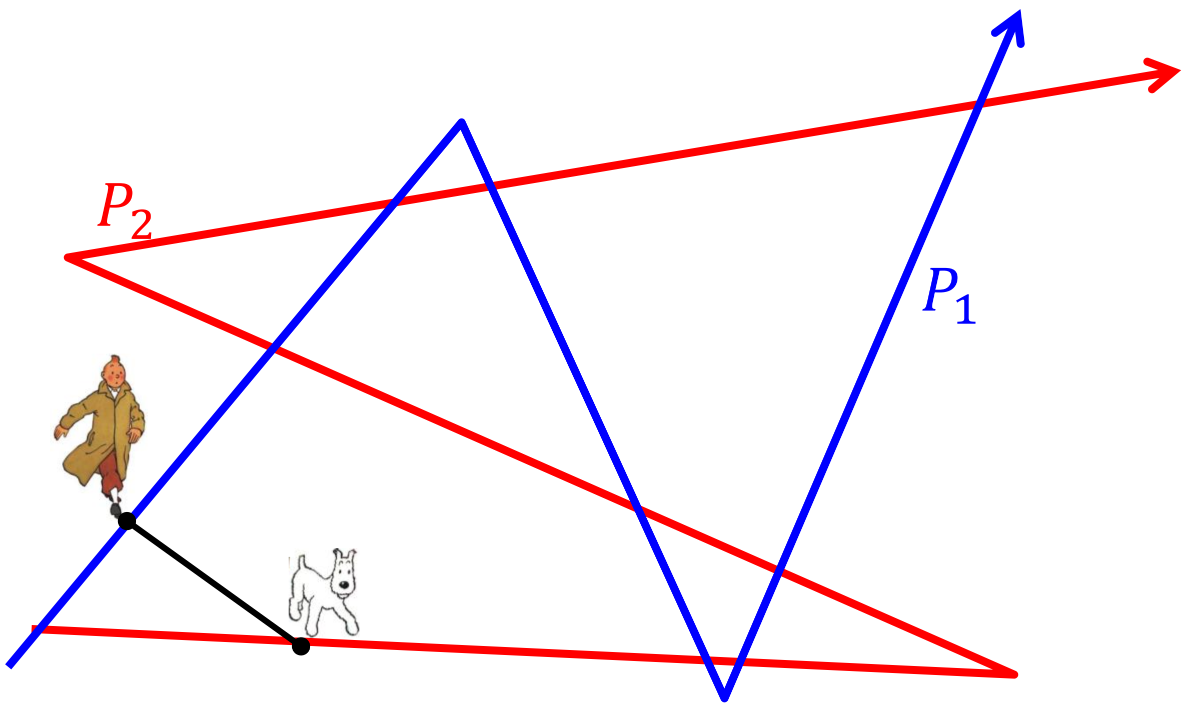


Fréchet-Distanz

Berechnung:

Löse Entscheidungsproblem: Ist $d_{\text{fréchet}} \leq \epsilon$?

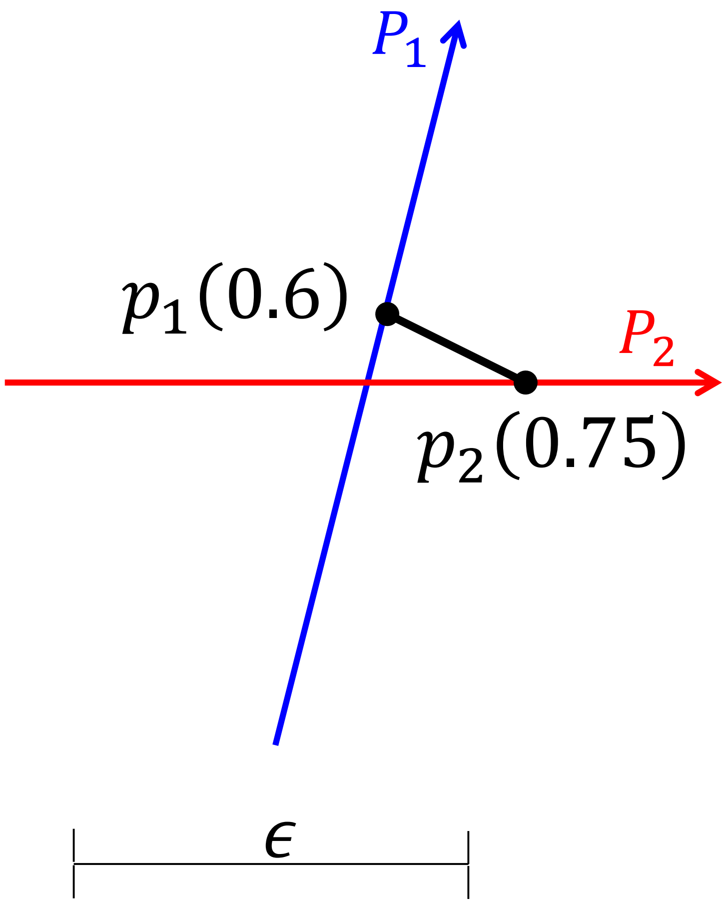
dann parametrische Suche...



Fréchet-Distanz

Ist $d_{\text{fréchet}} \leq \epsilon$?

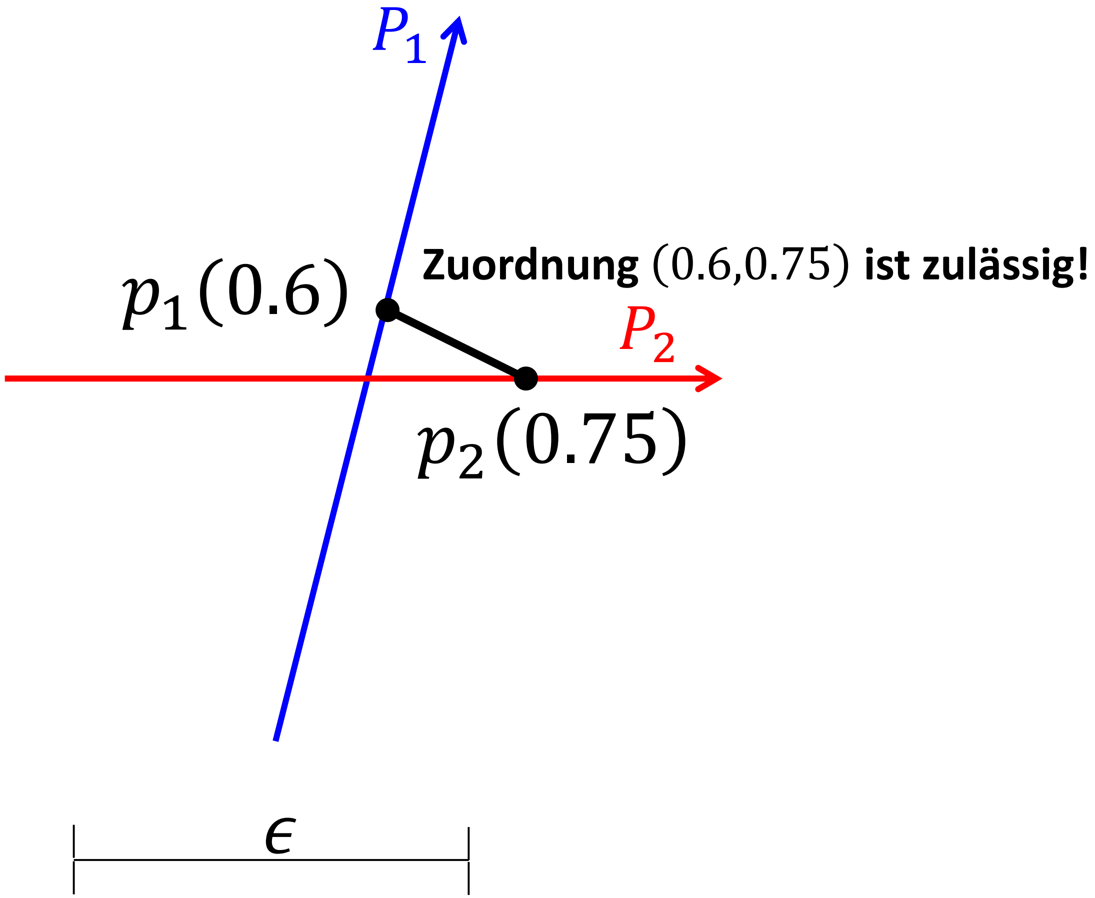
Spezialfall:



Fréchet-Distanz

Ist $d_{\text{fréchet}} \leq \epsilon$?

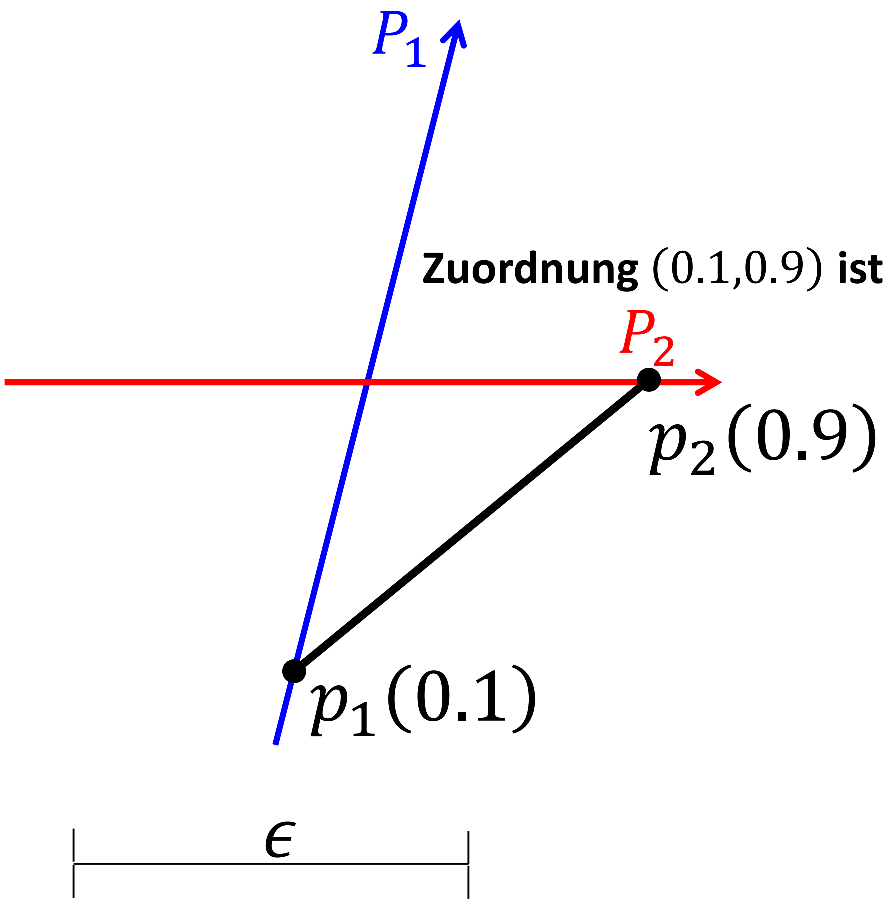
Spezialfall:



Fréchet-Distanz

Ist $d_{\text{fréchet}} \leq \epsilon$?

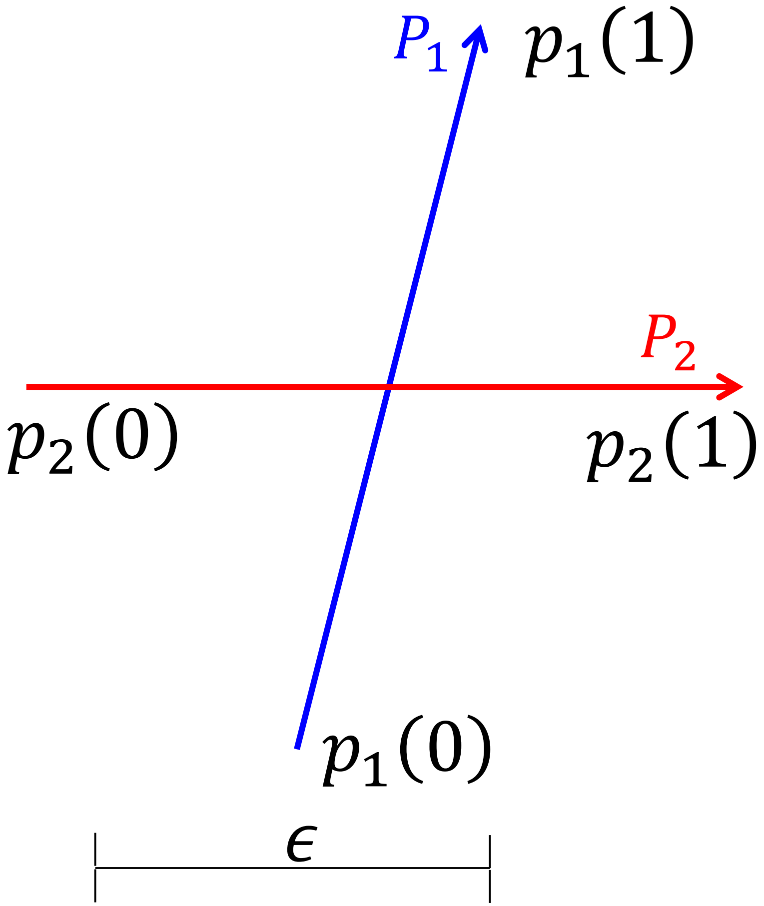
Spezialfall:

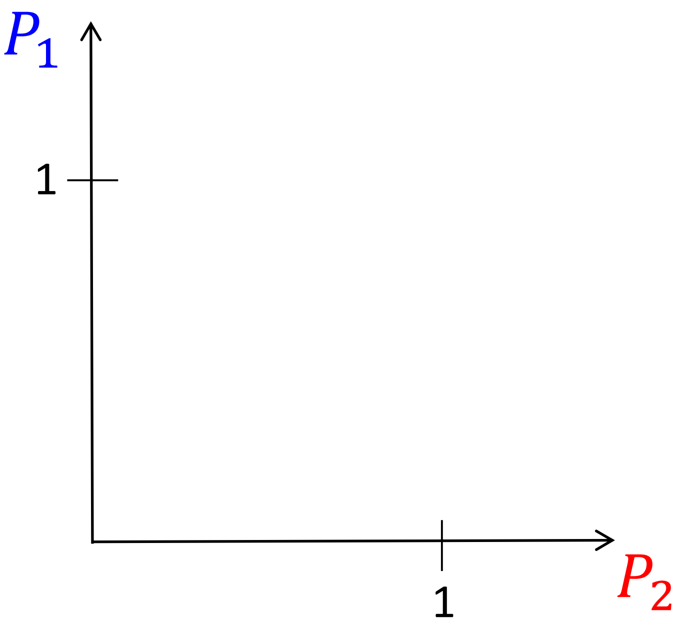


Fréchet-Distanz

Ist $d_{\text{fréchet}} \leq \epsilon$?

Spezialfall:

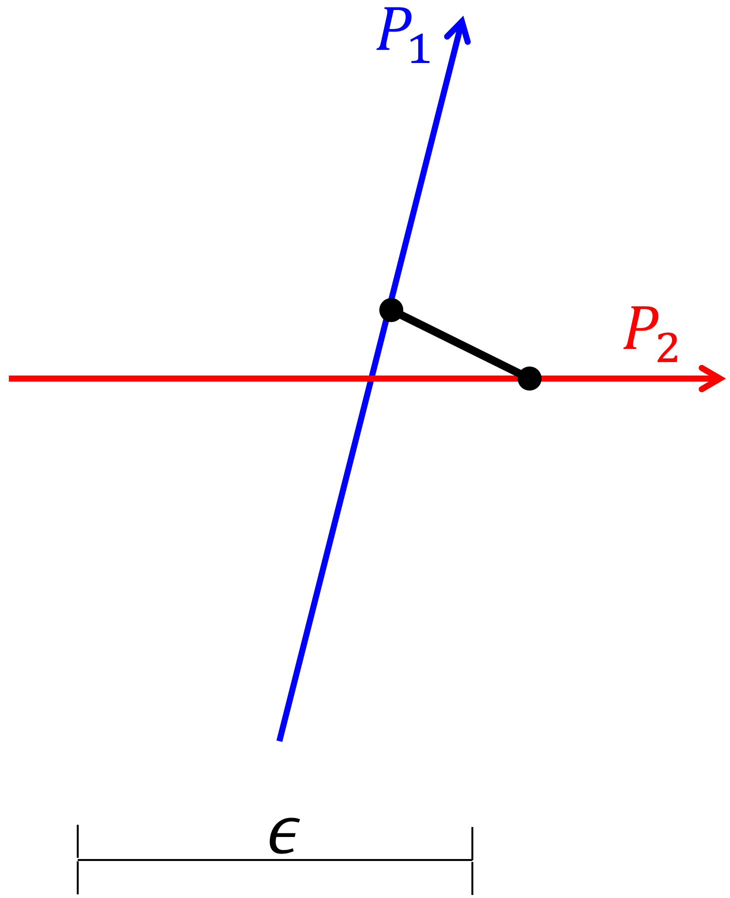




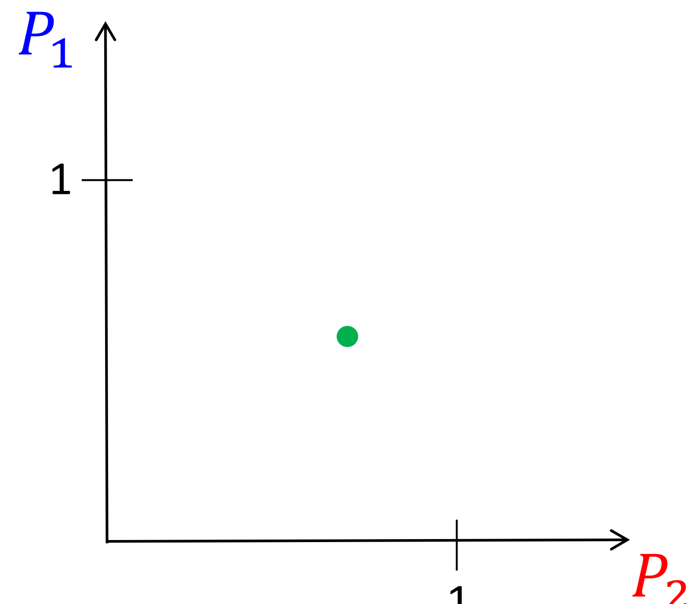
Fréchet-Distanz

Ist $d_{\text{fréchet}} \leq \epsilon$?

Spezialfall:



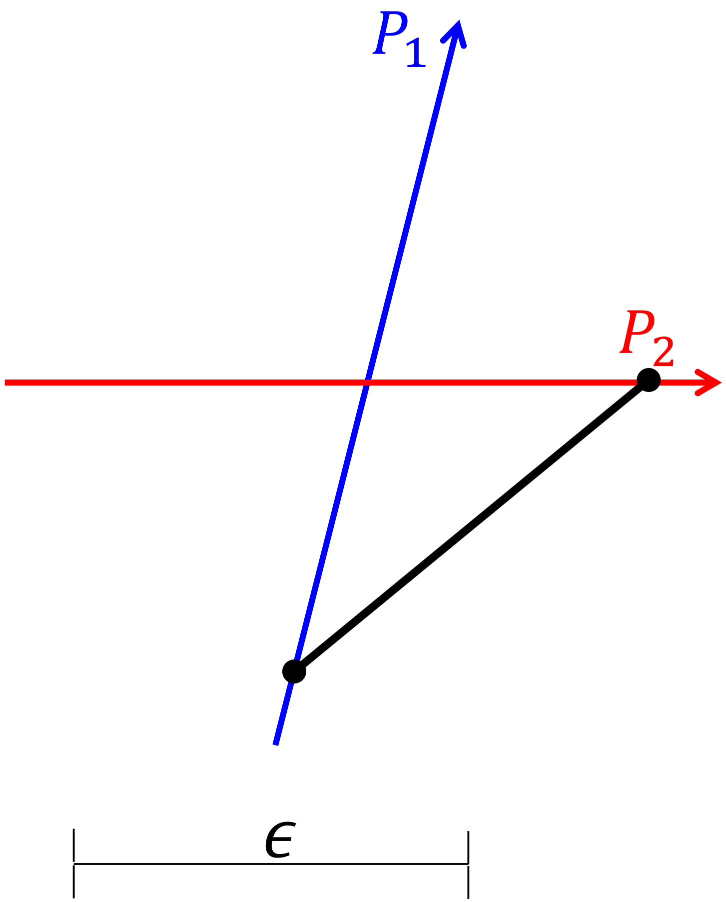
Zuordnung $(0.6, 0.75)$ **ist zulässig!**



Fréchet-Distanz

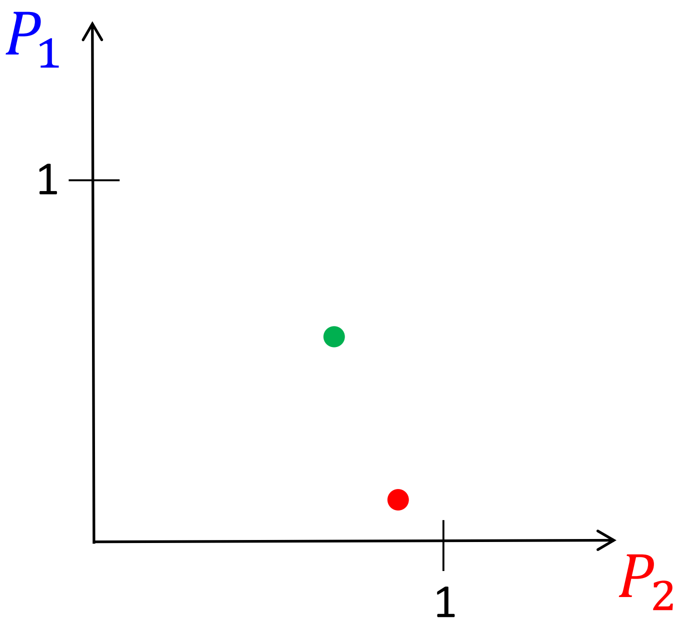
Ist $d_{\text{fréchet}} \leq \epsilon$?

Spezialfall:



Zuordnung $(0.6, 0.75)$ **ist zulässig!**

Zuordnung $(0.1, 0.9)$ **ist** ***nicht*** **zulässig!**

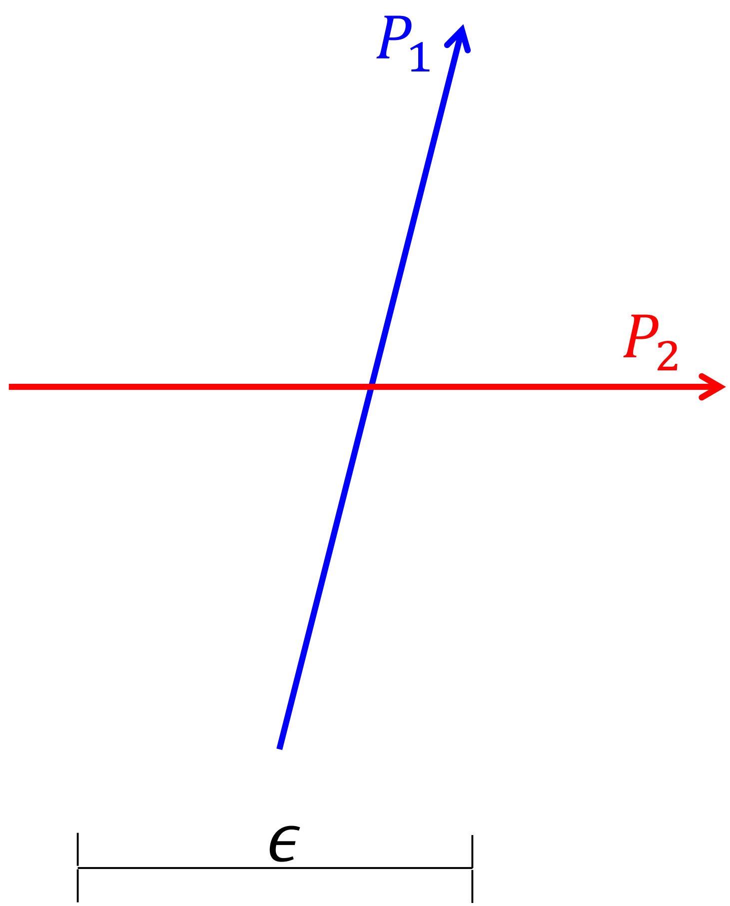


Fréchet-Distanz

Ist $d_{\text{fréchet}} \leq \epsilon$?

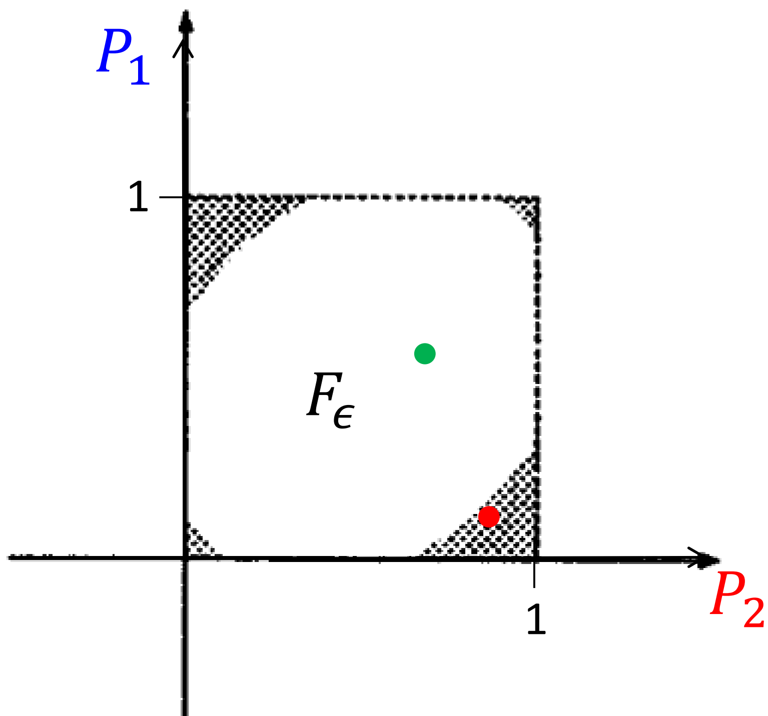
Spezialfall:

Freiraum
$F_\epsilon = \{(s,t) \in [0,1]^2 \mid$
hat die Form



Zuordnung $(0.6, 0.75)$ **ist zulässig!**

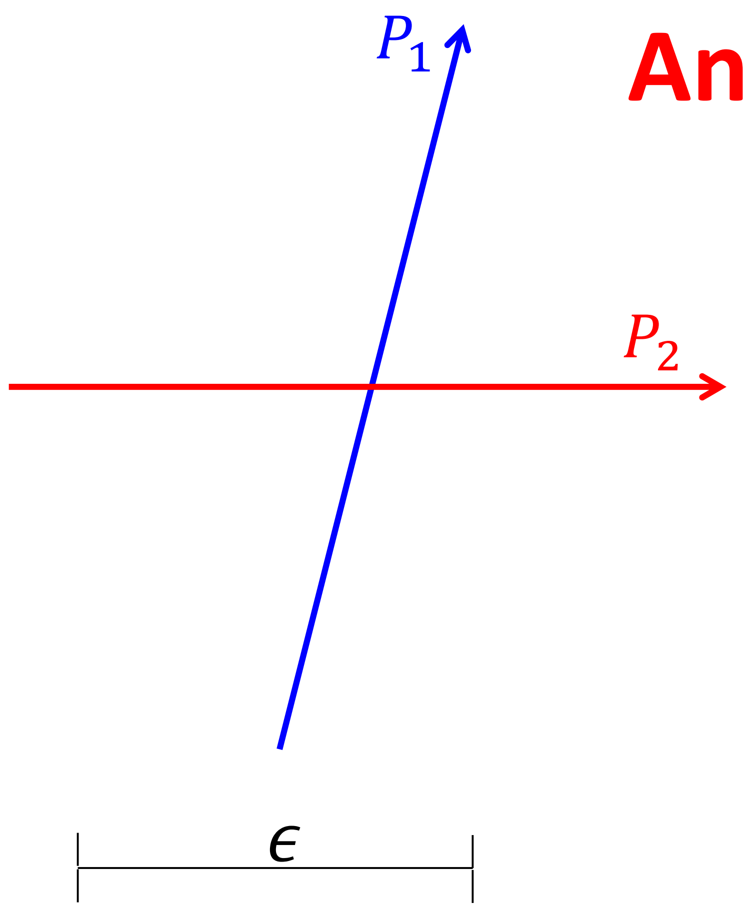
Zuordnung $(0.1, 0.9)$ **ist *nicht* zulässig!**



Fréchet-Distanz

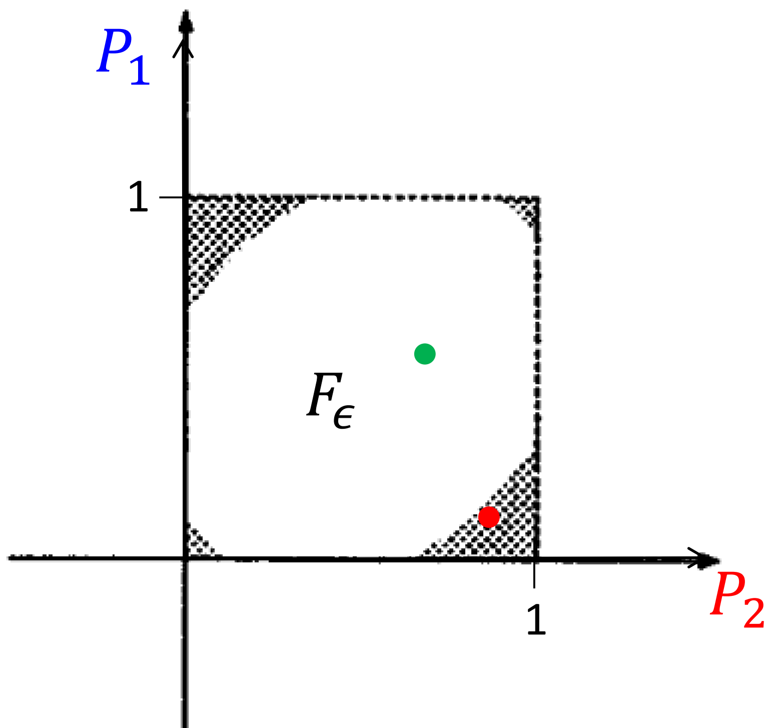
Ist $d_{\text{fréchet}} \leq \epsilon$?

Spezialfall:



Freiraum
$F_\epsilon = \{(s,t) \in [0,1]^2 \mid d\big(p_1(s), p_2(t)\big) \leq \epsilon\}$
hat die Form einer Ellipse.

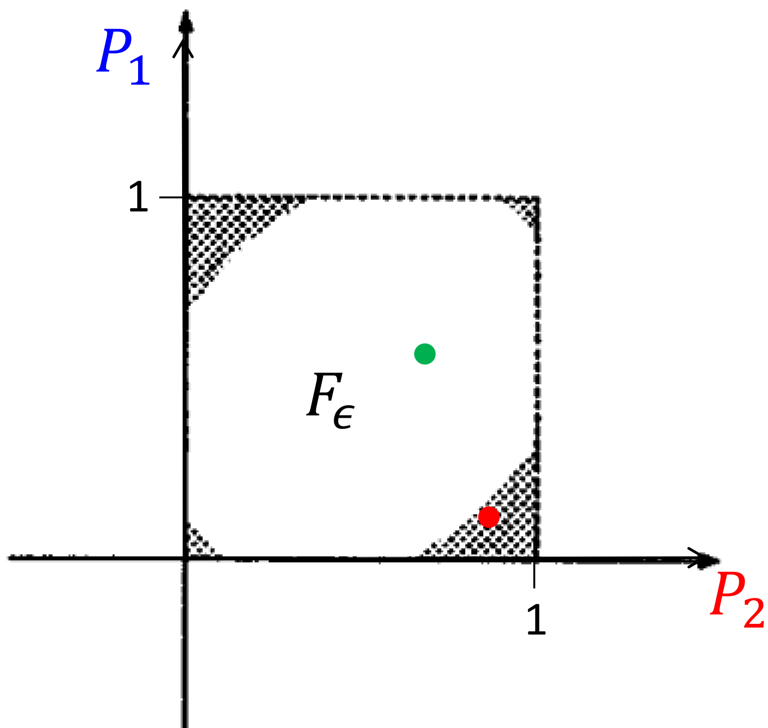
Anhand F_ϵ entscheiden!



Fréchet-Distanz

Freiraum
$F_\epsilon = \{(s,t) \in [0,1]^2 \mid d\big(p_1(s), p_2(t)\big) \le \epsilon\}$
hat die Form einer Ellipse.

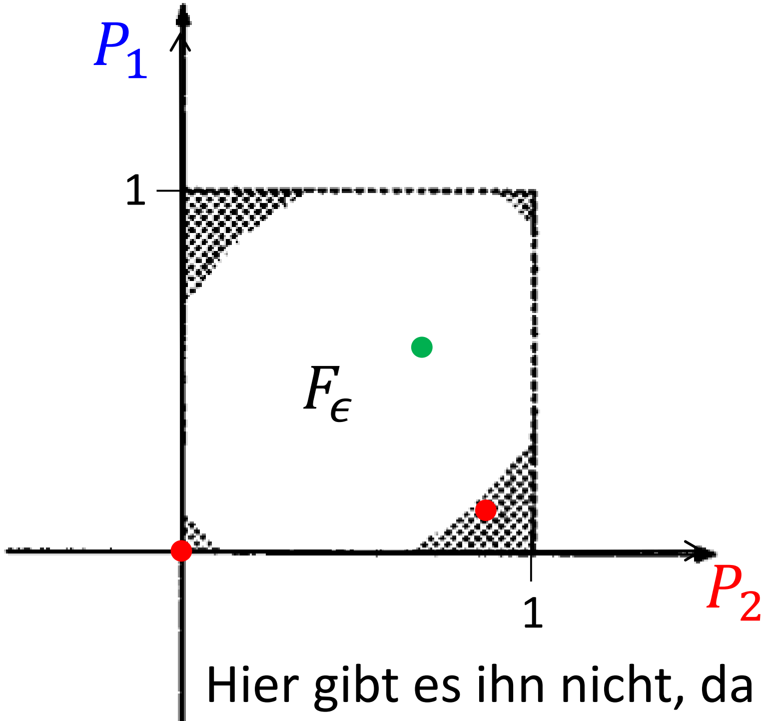
$d_{\text{fréchet}} \le \epsilon$
$\Leftrightarrow$ Es gibt einen Pfad
von nach
im Inneren von F_ϵ,



Fréchet-Distanz

Freiraum
$F_\epsilon = \{(s,t) \in [0,1]^2 \mid d\big(p_1(s), p_2(t)\big) \leq \epsilon\}$
hat die Form einer Ellipse.

$d_{\text{fréchet}} \leq \epsilon$
$\Leftrightarrow$ Es gibt einen Pfad
von $(0,0)$ nach $(1,1)$
im Inneren von F_ϵ,
der in beide Richtungen
monoton ist.

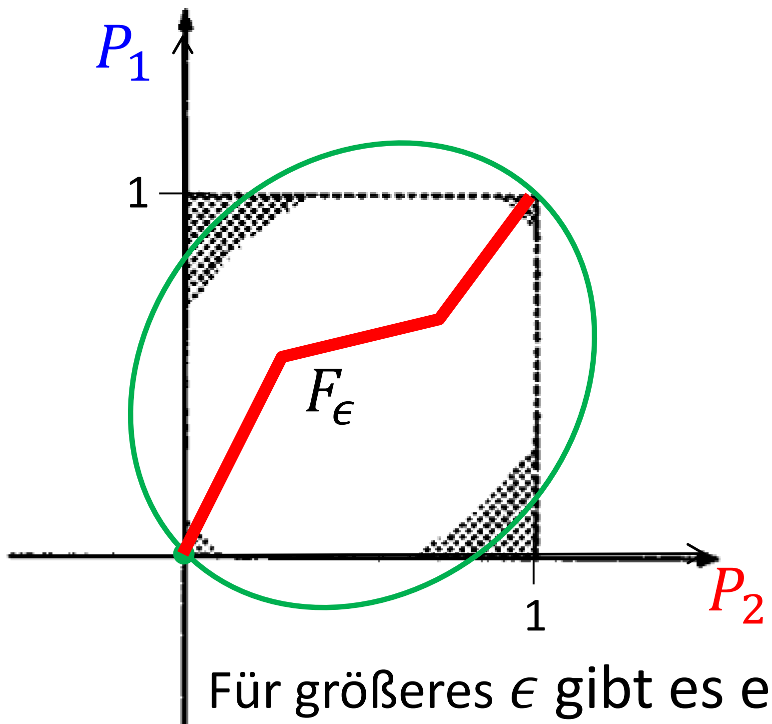


Hier gibt es ihn nicht, da $(0,0) \notin F_\epsilon$!

Fréchet-Distanz

Freiraum
$F_\epsilon = \{(s,t) \in [0,1]^2 \mid d(p_1(s), p_2(t)) \leq \epsilon\}$
hat die Form einer Ellipse.

$d_{\text{fréchet}} \leq \epsilon$
$\Leftrightarrow$ Es gibt einen Pfad
von $(0,0)$ nach $(1,1)$
im Inneren von F_ϵ,
der in beide Richtungen
monoton ist.

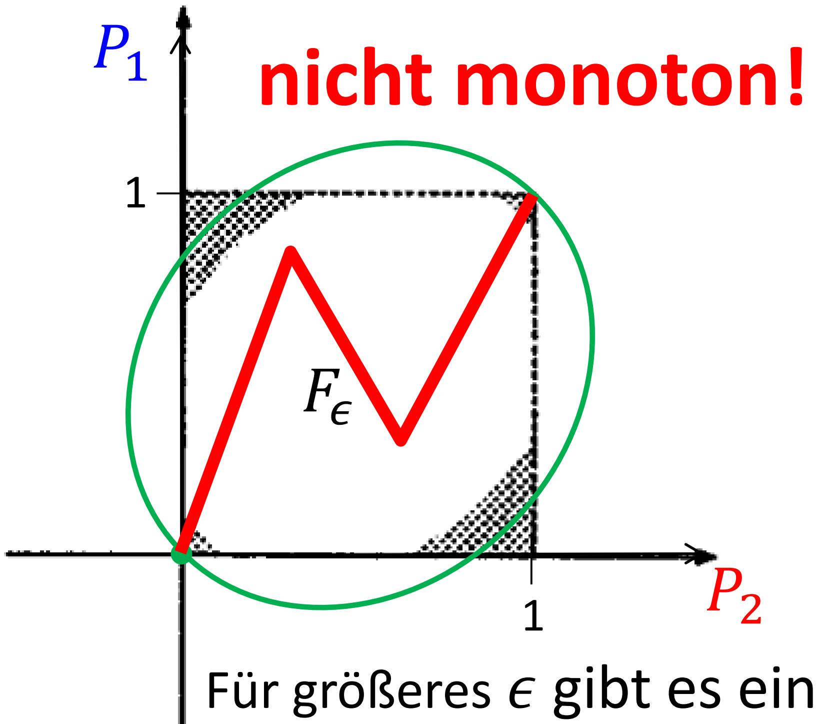


Für größeres ϵ gibt es einen Pfad!

Fréchet-Distanz

Freiraum
$F_\epsilon = \{(s,t) \in [0,1]^2 \mid d(p_1(s), p_2(t)) \leq \epsilon\}$
hat die Form einer Ellipse.

$d_{\text{fréchet}} \leq \epsilon$
$\Leftrightarrow$ Es gibt einen Pfad von $(0,0)$ nach $(1,1)$ im Inneren von F_ϵ, der in beide Richtungen monoton ist.



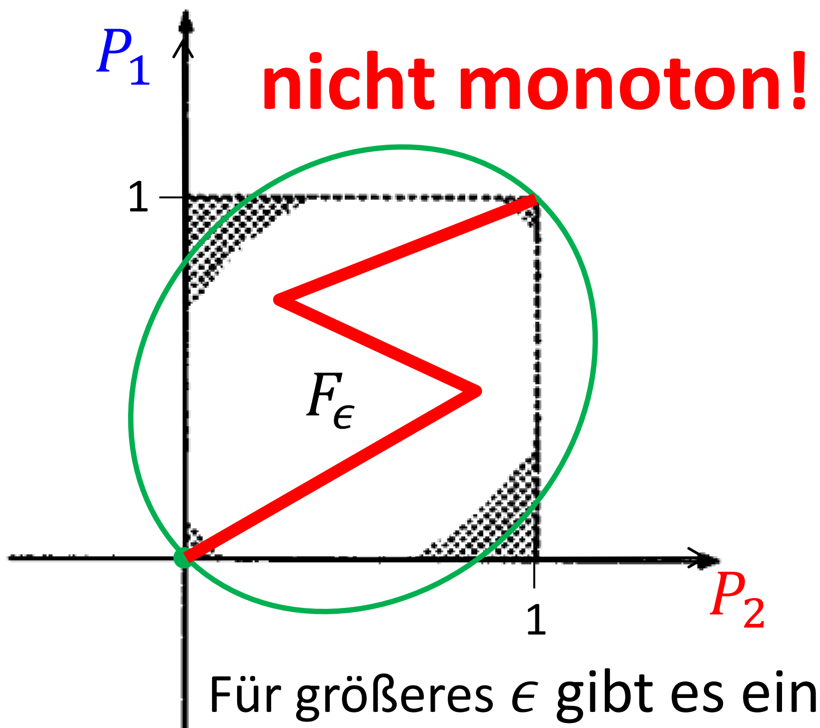
Für größeres ϵ gibt es einen Pfad!

Fréchet-Distanz

Freiraum
$F_\epsilon = \{(s,t) \in [0,1]^2 \mid d(p_1(s), p_2(t)) \leq \epsilon\}$
hat die Form einer Ellipse.

mehr als 1 Segment?

$d_{\text{fréchet}} \leq \epsilon$
$\Leftrightarrow$ Es gibt einen Pfad
von $(0,0)$ nach $(1,1)$
im Inneren von F_ϵ,
der in beide Richtungen
monoton ist.

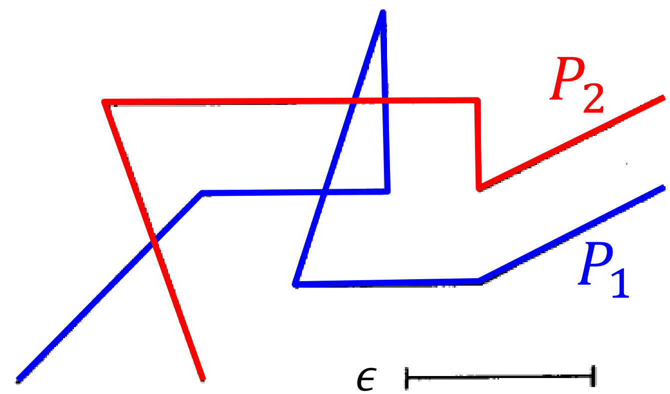


Für größeres ϵ gibt es einen Pfad!

Fréchet-Distanz

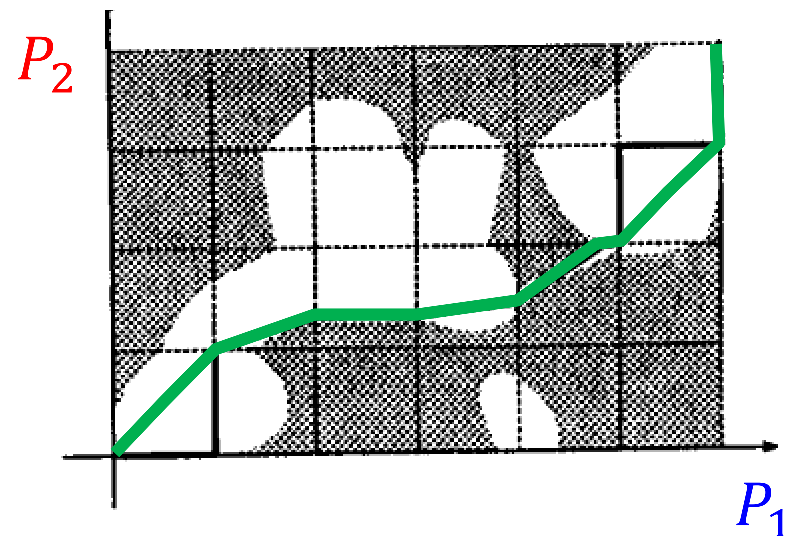
Freiraum

$$F_\epsilon = \{(s,t) \in [0, m-1] \times [0, n-1] \mid d(p_1(s), p_2(t)) \leq \epsilon\}$$



$d_{\text{fréchet}} \leq \epsilon$

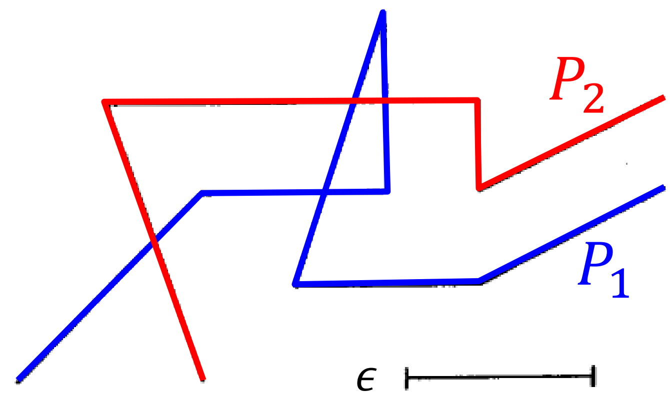
$\Leftrightarrow$ Es gibt einen Pfad von $(0{,}0)$ nach (m, n) im Inneren von F_ϵ, der in beide Richtungen monoton ist.



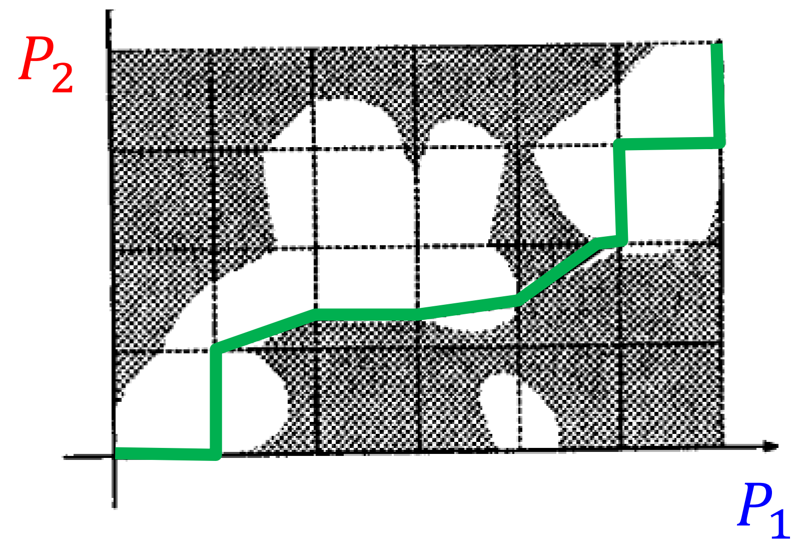
Fréchet-Distanz

Freiraum

$$F_\epsilon = \{(s,t) \in [0, m-1] \times [0, n-1] \mid d(p_1(s), p_2(t)) \le \epsilon\}$$



$d_{\text{fréchet}} \le \epsilon$
$\Leftrightarrow$ Es gibt einen Pfad
von $(0,0)$ nach (m, n)
im Inneren von F_ϵ,
der in beide Richtungen
monoton ist.



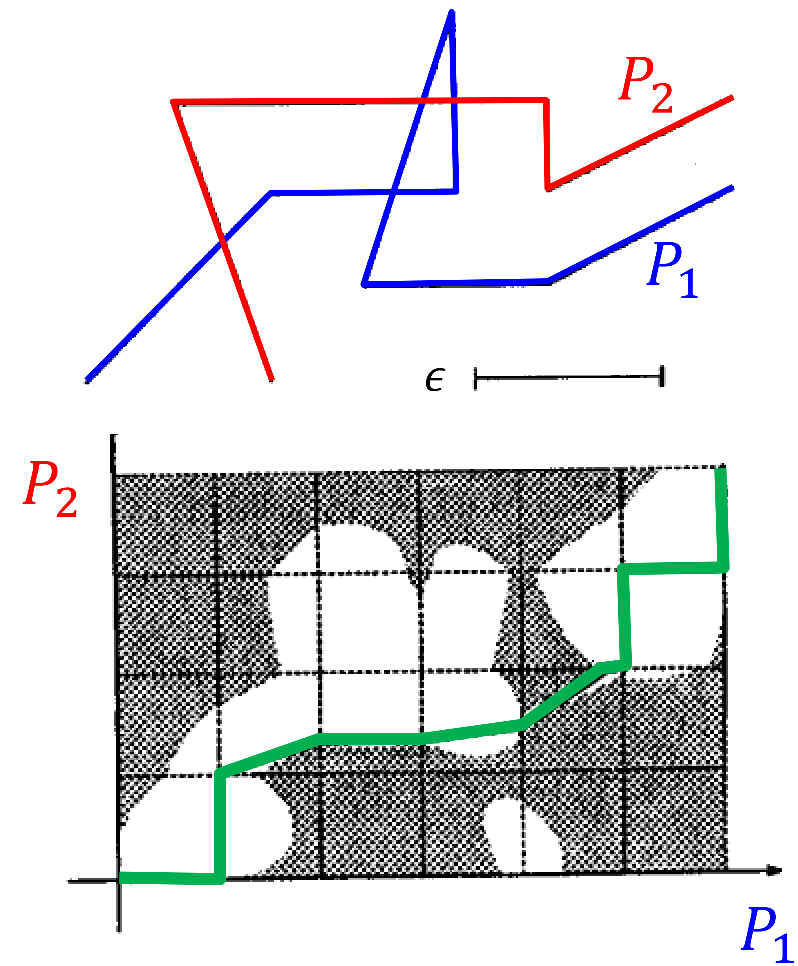
Fréchet-Distanz

Freiraum

$$F_\epsilon = \{(s,t) \in [0, m-1] \times [0, n-1] \mid d(p_1(s), p_2(t)) \le \epsilon\}$$

$d_{\text{fréchet}} \le \epsilon$
$\Leftrightarrow$ Es gibt einen Pfad von $(0{,}0)$ nach (m, n) im inneren von F_ϵ, der in beide Richtungen monoton ist.

F_ϵ kann in $O(mn)$ Zeit berechnet werden.



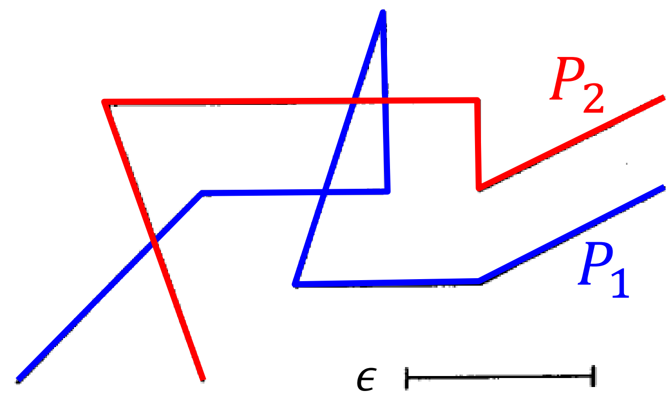
Fréchet-Distanz

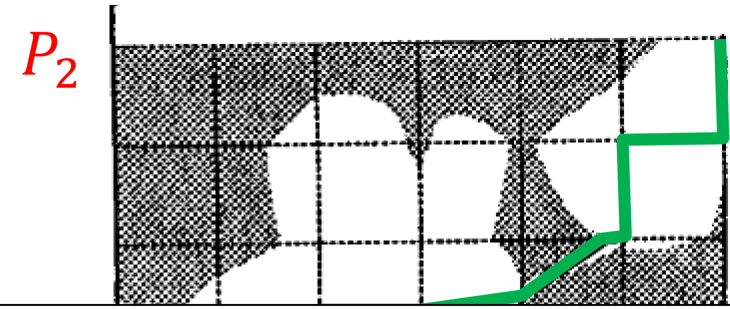
Freiraum

$$F_\epsilon = \{(s,t) \in [0, m-1] \times [0, n-1] \mid d(p_1(s), p_2(t)) \leq \epsilon\}$$

$d_{\text{fréchet}} \leq \epsilon$

$\Leftrightarrow$ Es gibt einen Pfad von $(0{,}0)$ nach (m, n) im inneren von F_ϵ, der in beide Richtungen monoton ist.





F_ϵ kann in $O(mn)$ Zeit berechnet werden.

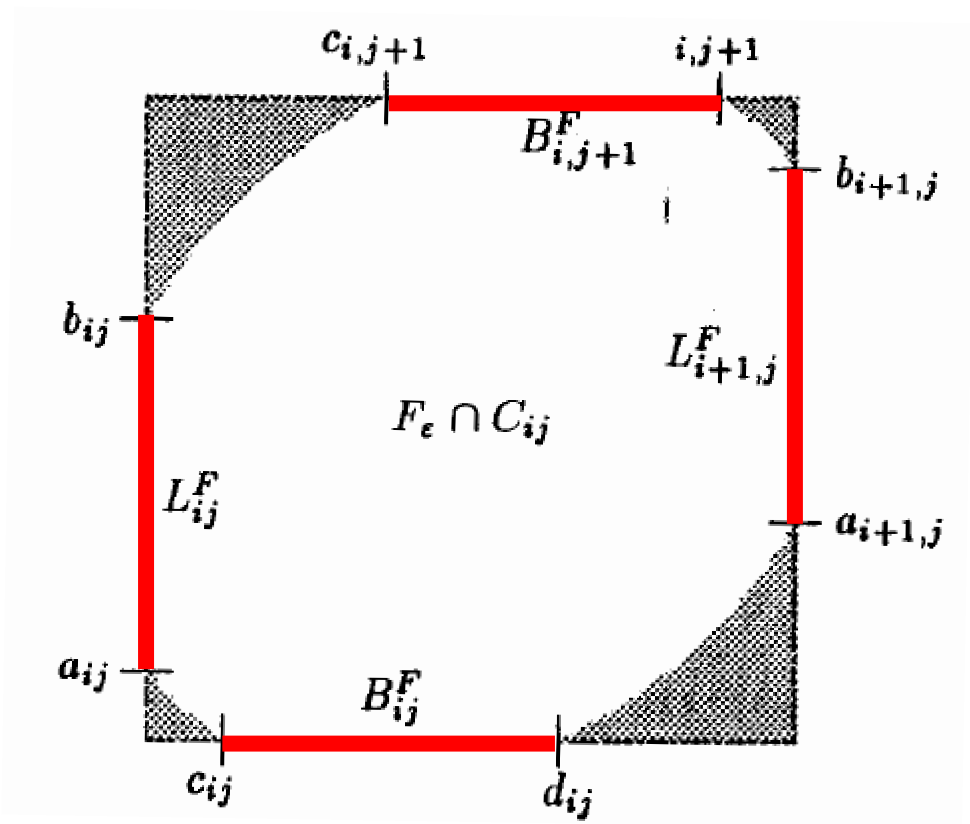
TODO:

Finde zulässigen Pfad in F_ϵ.

Fréchet-Distanz

Kanten einer Freiraumzelle:

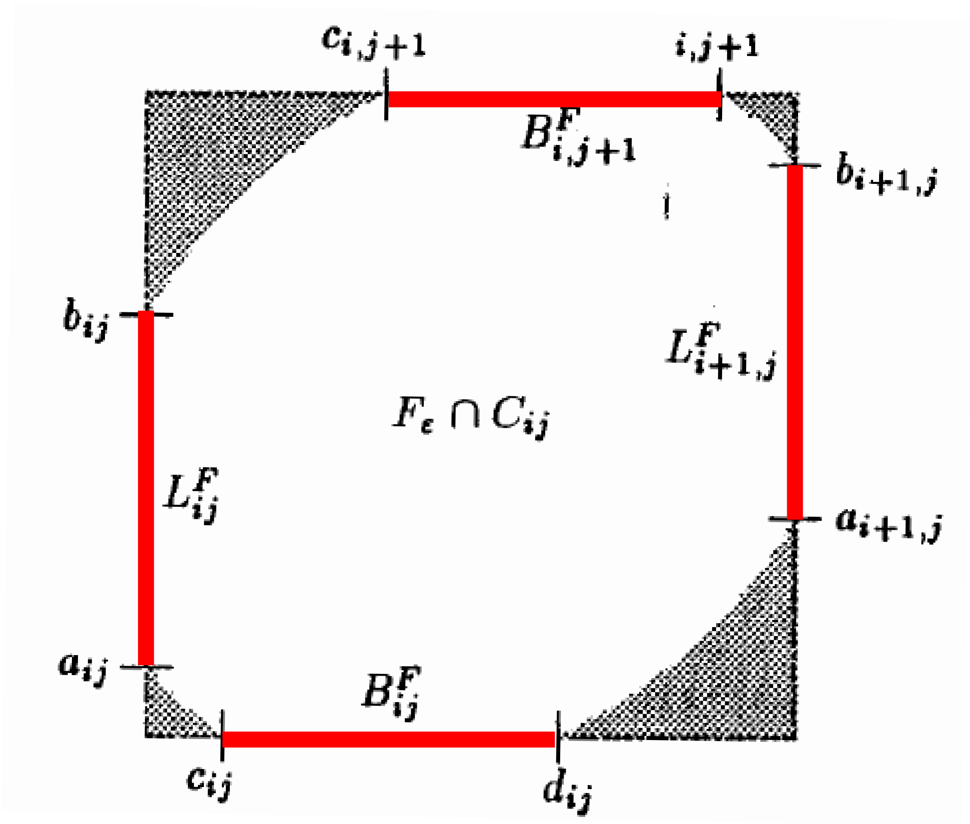
$B_{i,j}^F, B_{i,j+1}^F, L_{i,j}^F, L_{i+1,j}^F$



Fréchet-Distanz

Kanten einer Freiraumzelle:

$B^F_{i,j}, B^F_{i,j+1}, L^F_{i,j}, L^F_{i+1,j}$



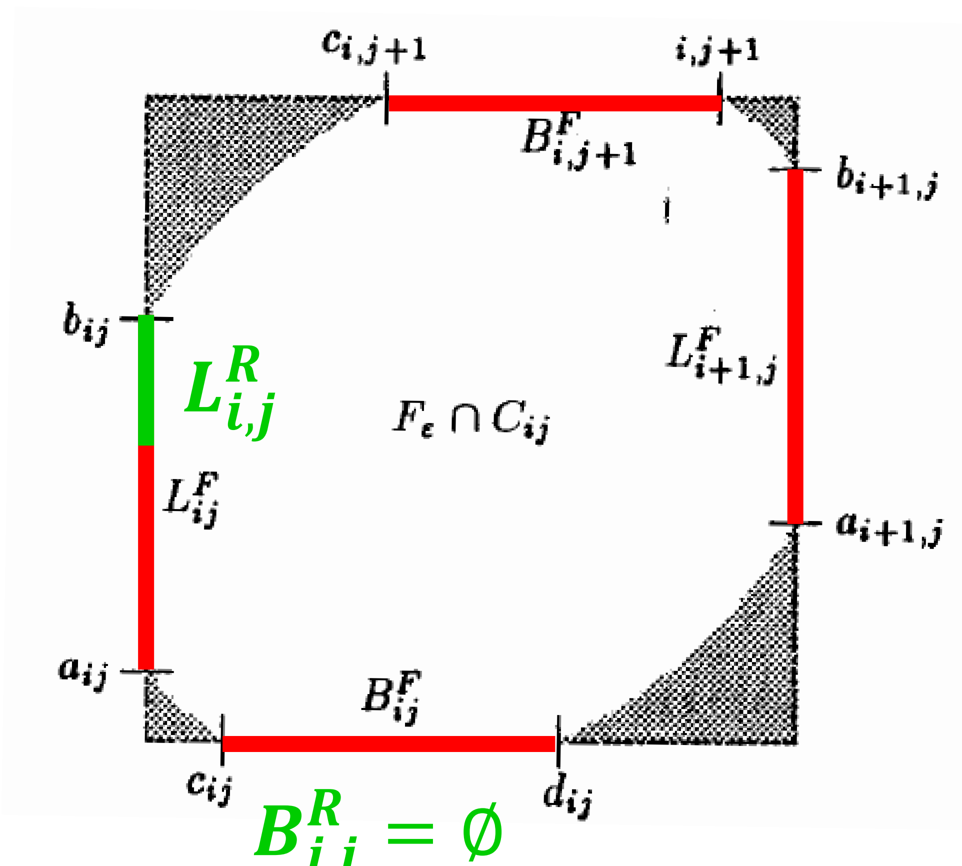
Teile dieser Kanten, die von $(0,0)$ aus erreichbar sind:

$B^R_{i,j}, B^R_{i,j+1}, L^R_{i,j}\ , L^R_{i+1,j}$

Fréchet-Distanz

Kanten einer Freiraumzelle:

$B^F_{i,j}, B^F_{i,j+1}, L^F_{i,j}, L^F_{i+1,j}$



Teile dieser Kanten, die von $(0,0)$ aus erreichbar sind:

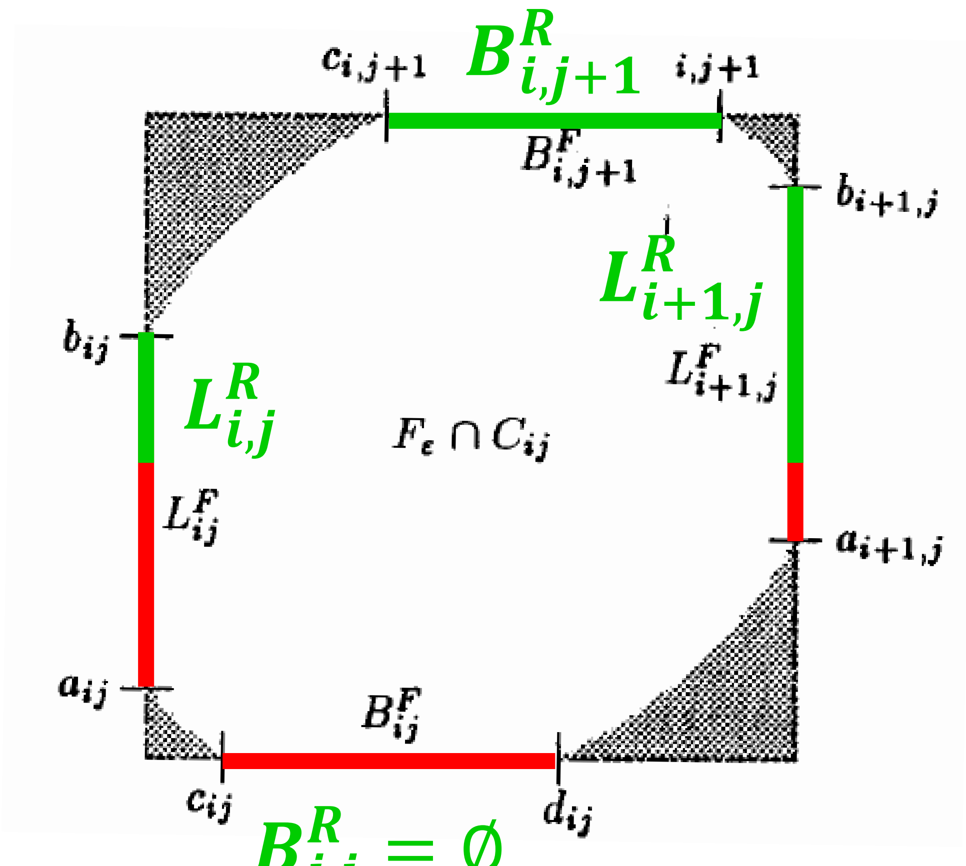
$B^R_{i,j}, B^R_{i,j+1}, L^R_{i,j}, L^R_{i+1,j}$

$B^R_{i,j}, L^R_{i,j}$ bekannt

Fréchet-Distanz

Kanten einer Freiraumzelle:

$B^F_{i,j}, B^F_{i,j+1}, L^F_{i,j}, L^F_{i+1,j}$



Teile dieser Kanten, die von $(0,0)$ aus erreichbar sind:

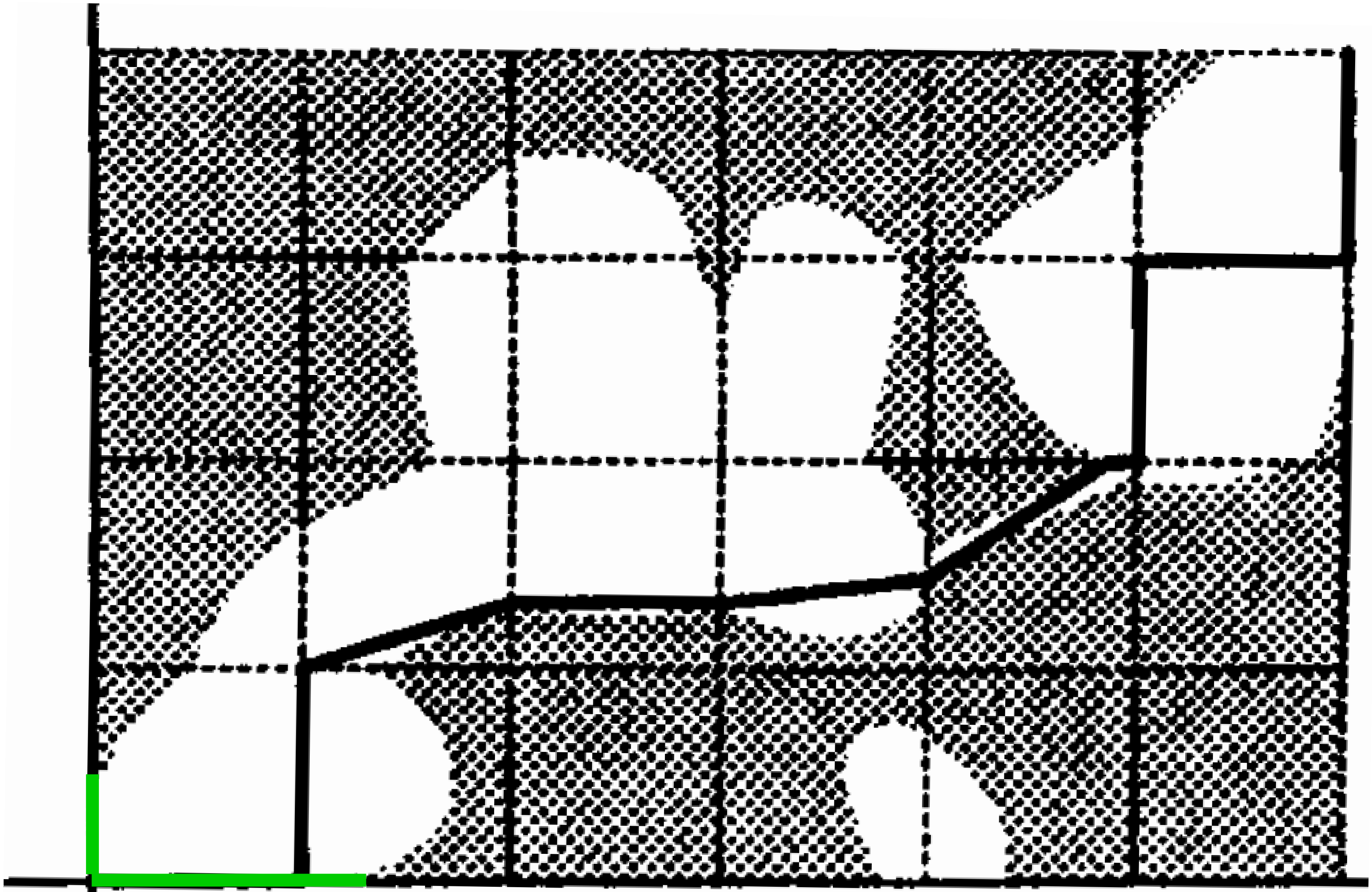
$B^R_{i,j}, B^R_{i,j+1}, L^R_{i,j}\ , L^R_{i+1,j}$

$B^R_{i,j}, L^R_{i,j}$ bekannt

$\Rightarrow B^R_{i,j+1}, L^R_{i+1,j}$

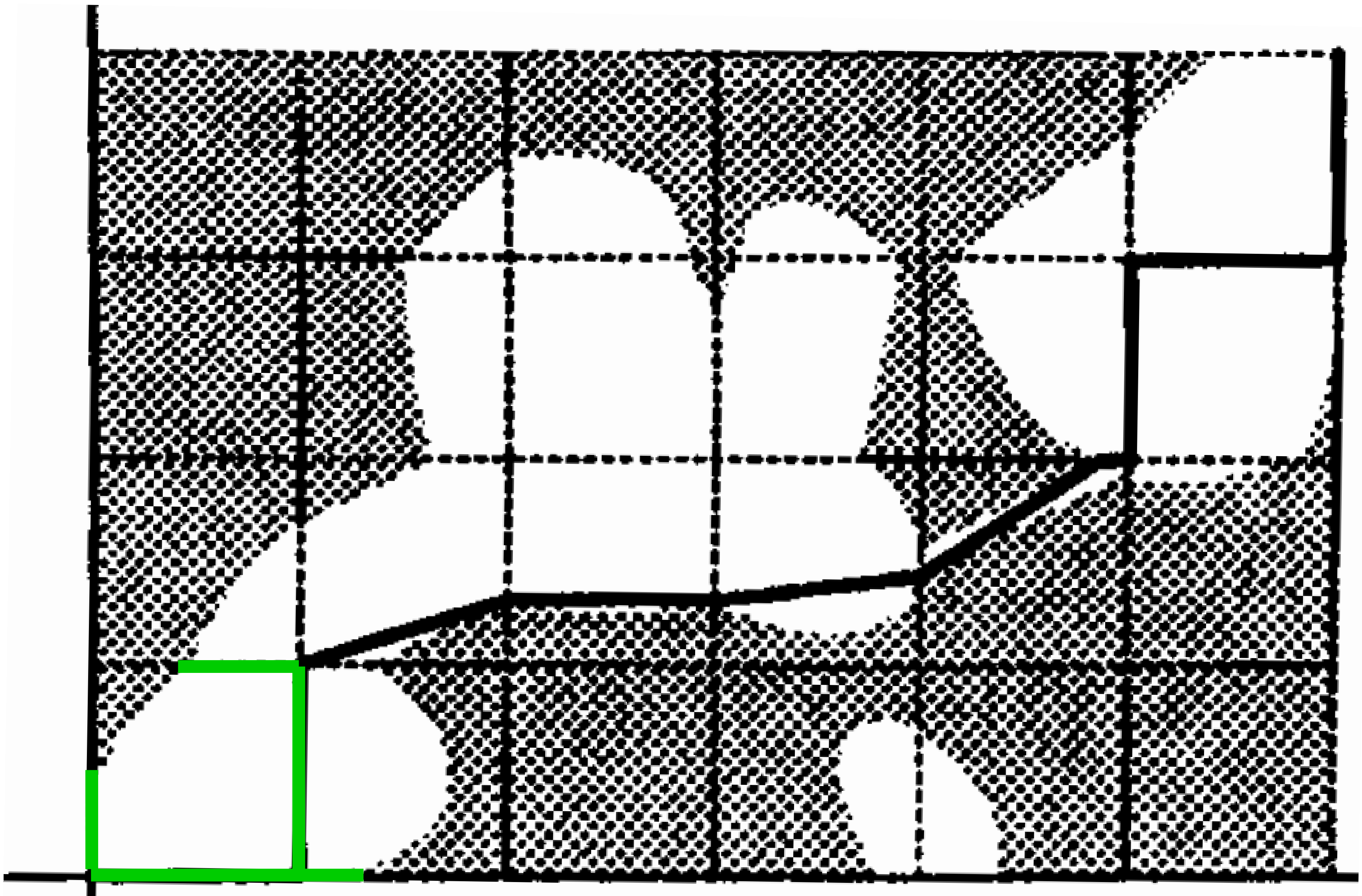
Fréchet-Distanz

Algorithmus



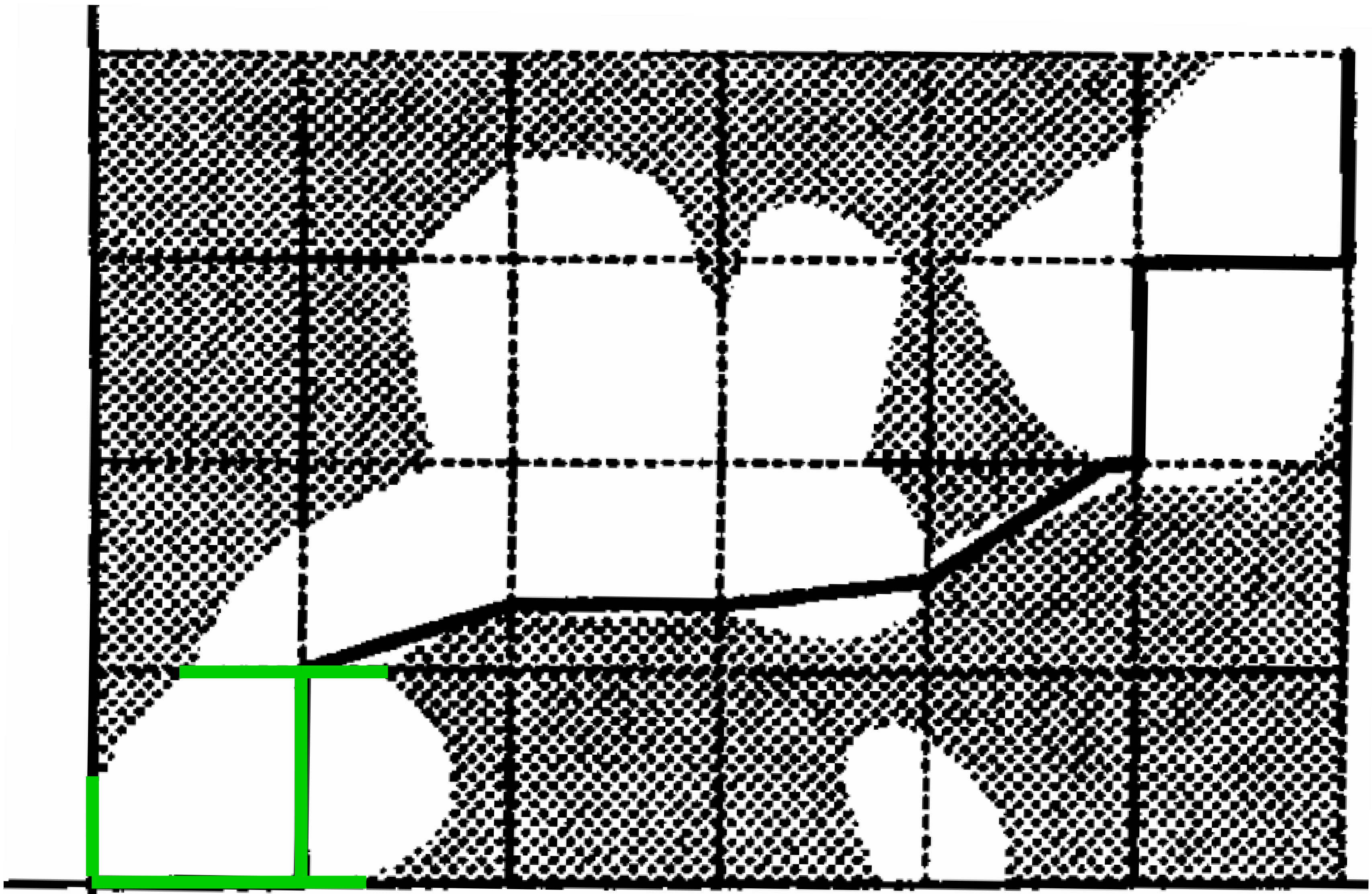
Fréchet-Distanz

Algorithmus



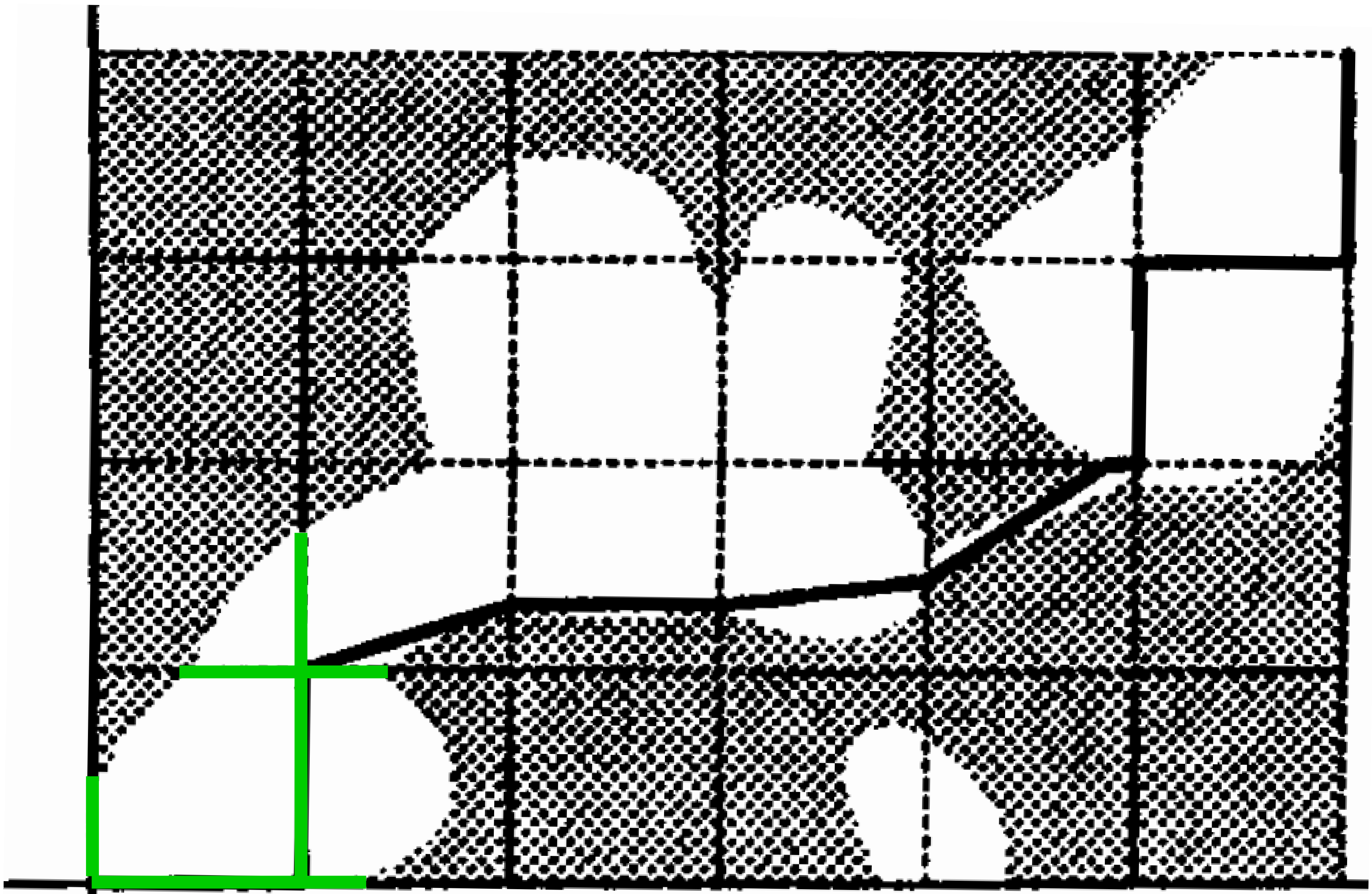
Fréchet-Distanz

Algorithmus



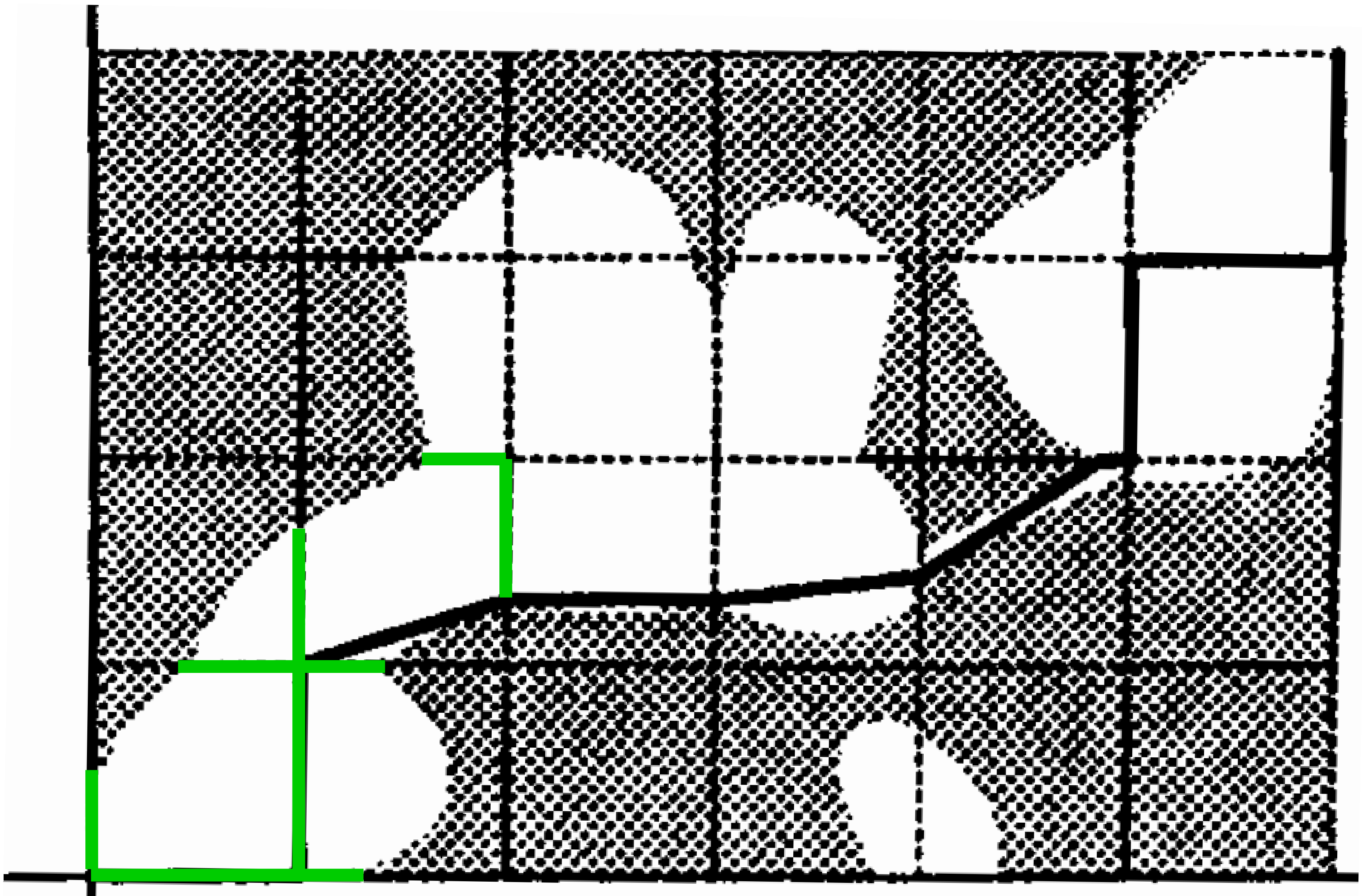
Fréchet-Distanz

Algorithmus



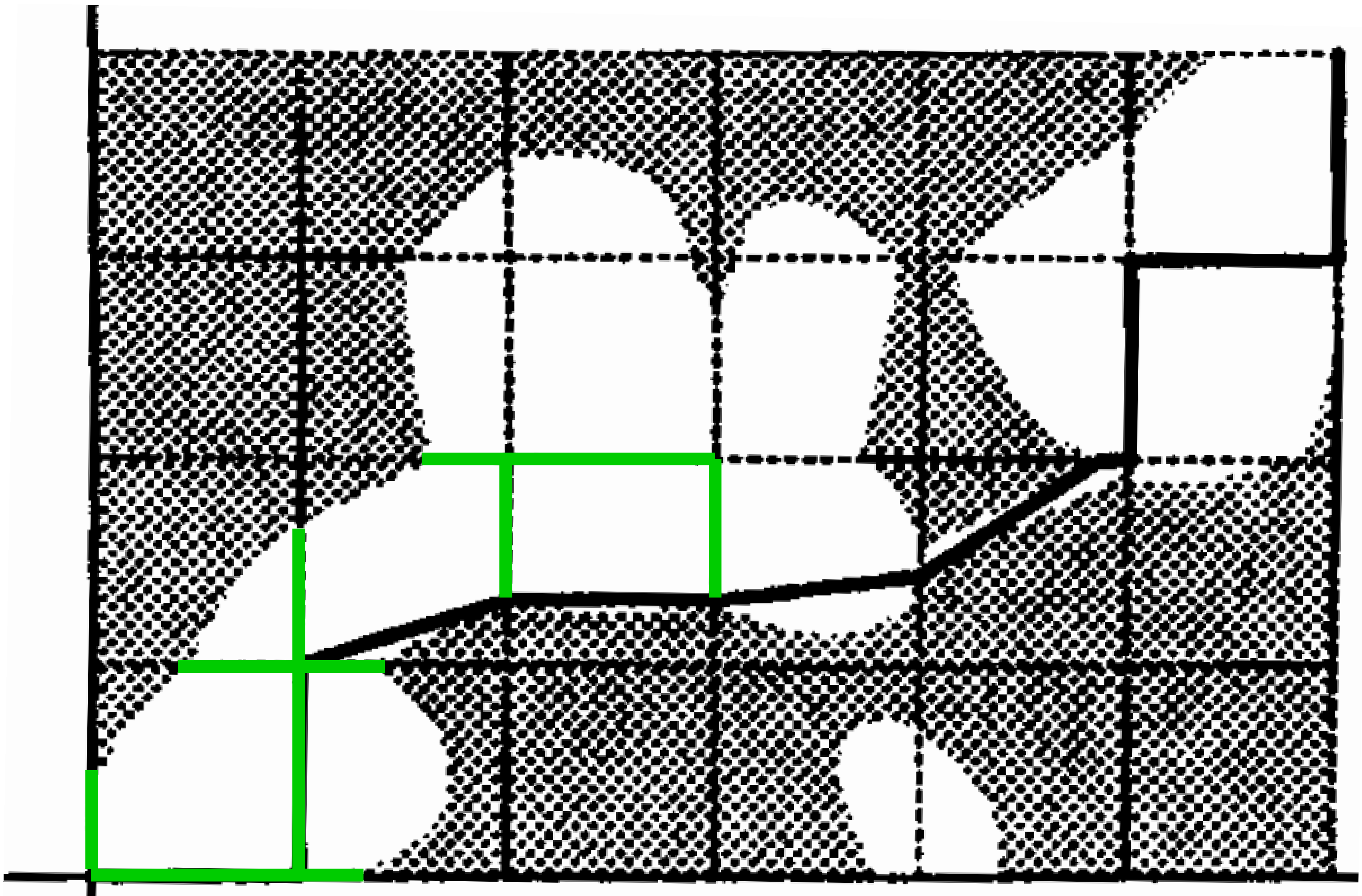
Fréchet-Distanz

Algorithmus



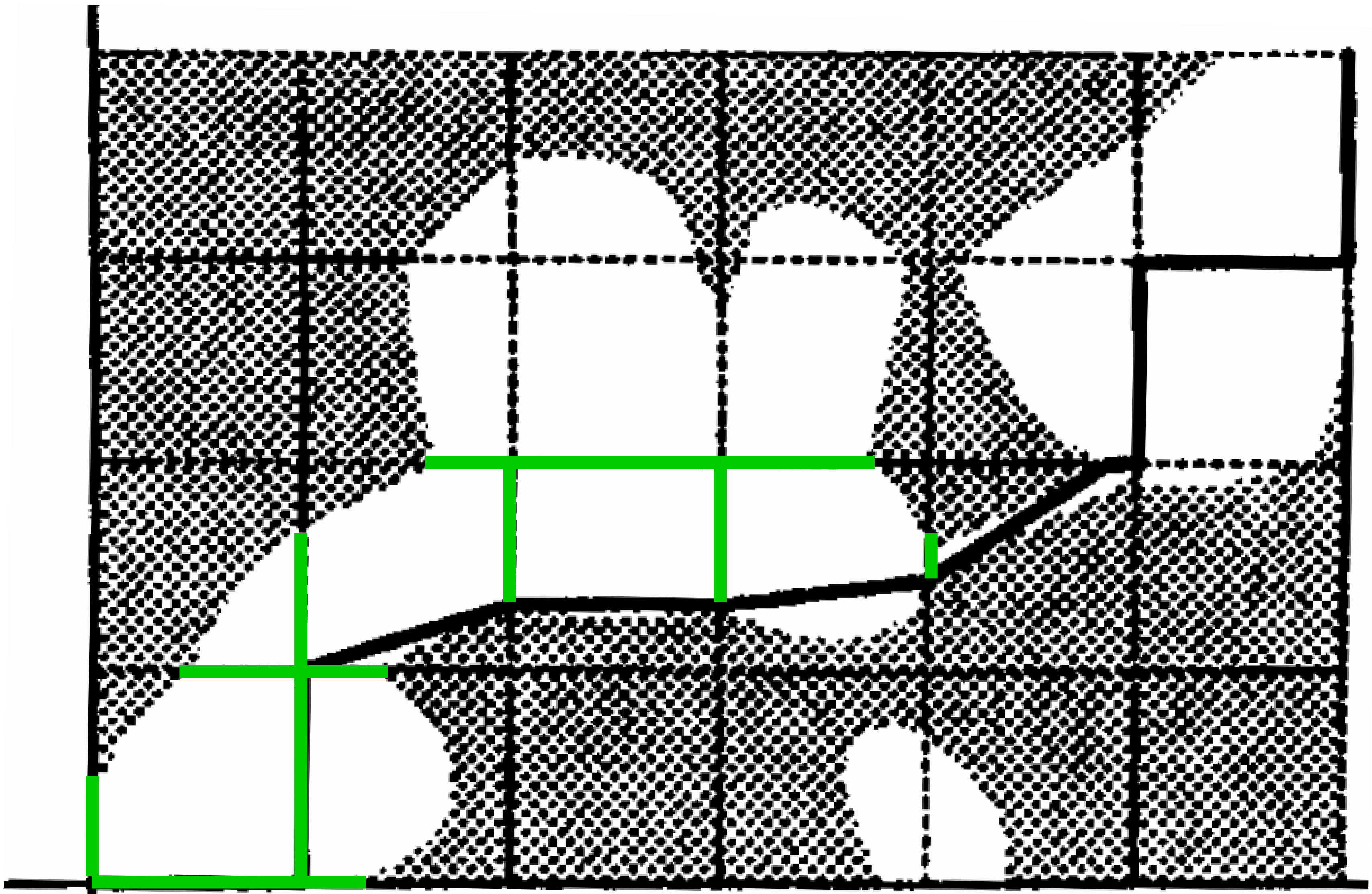
Fréchet-Distanz

Algorithmus



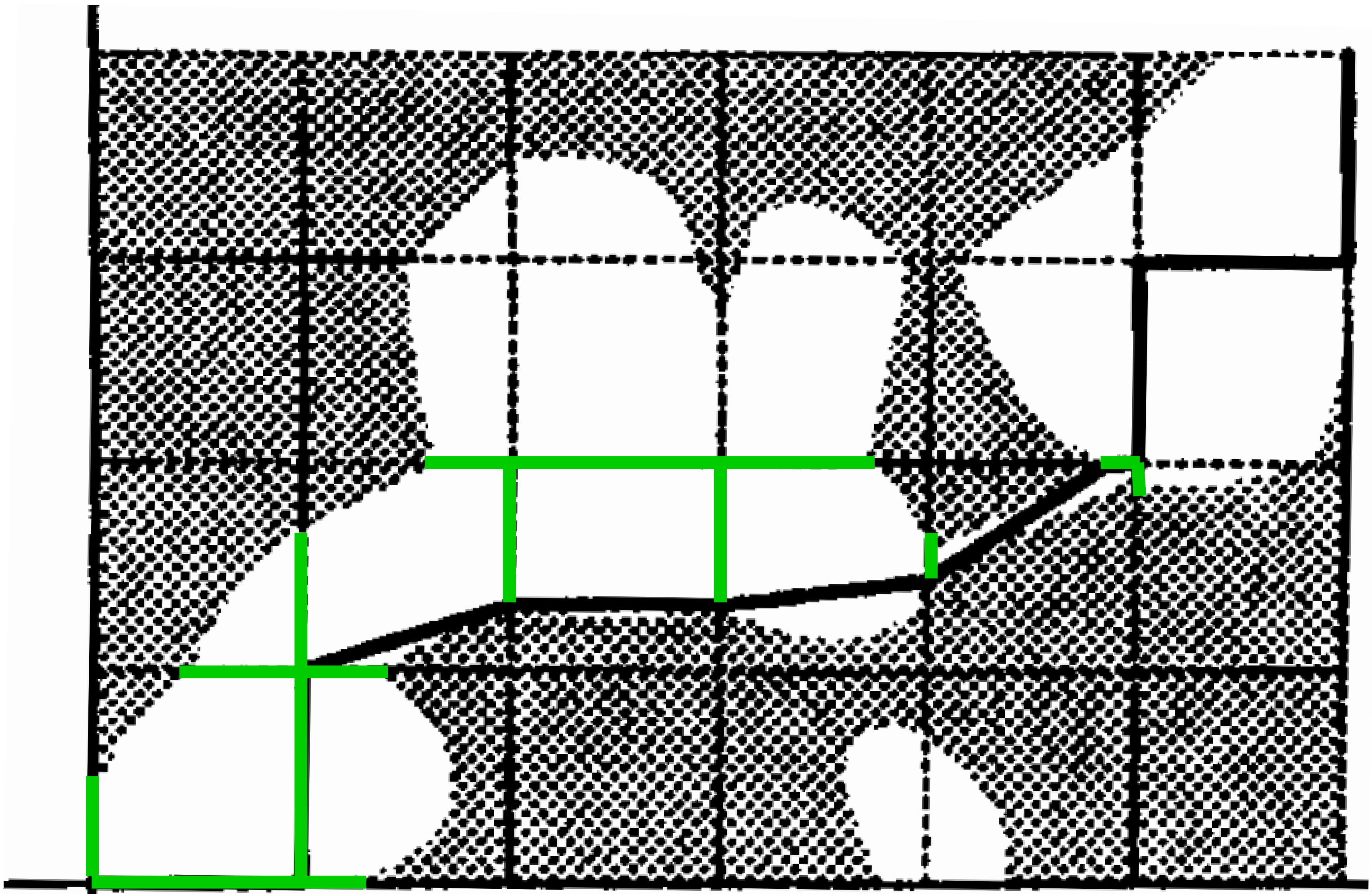
Fréchet-Distanz

Algorithmus



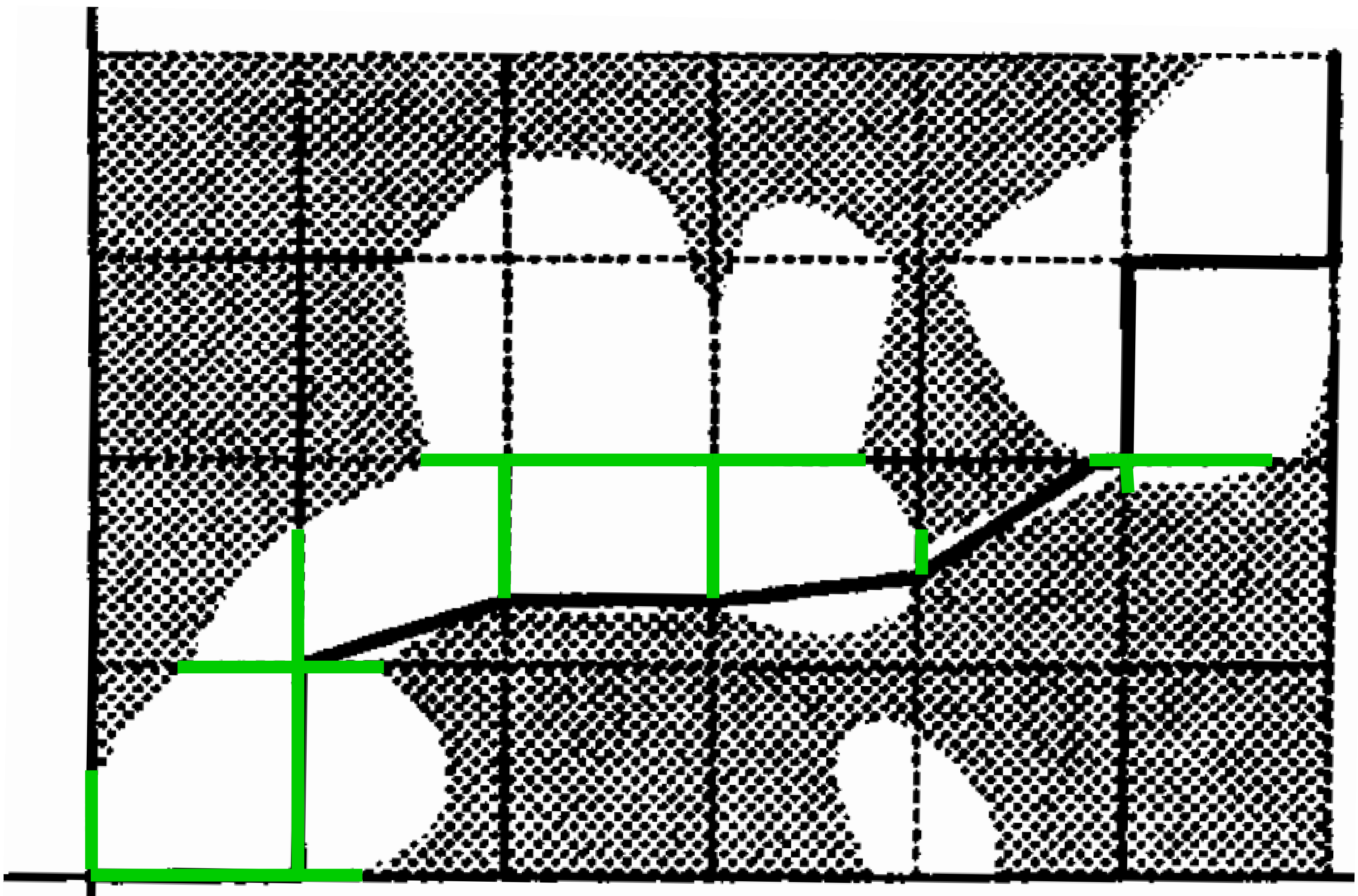
Fréchet-Distanz

Algorithmus



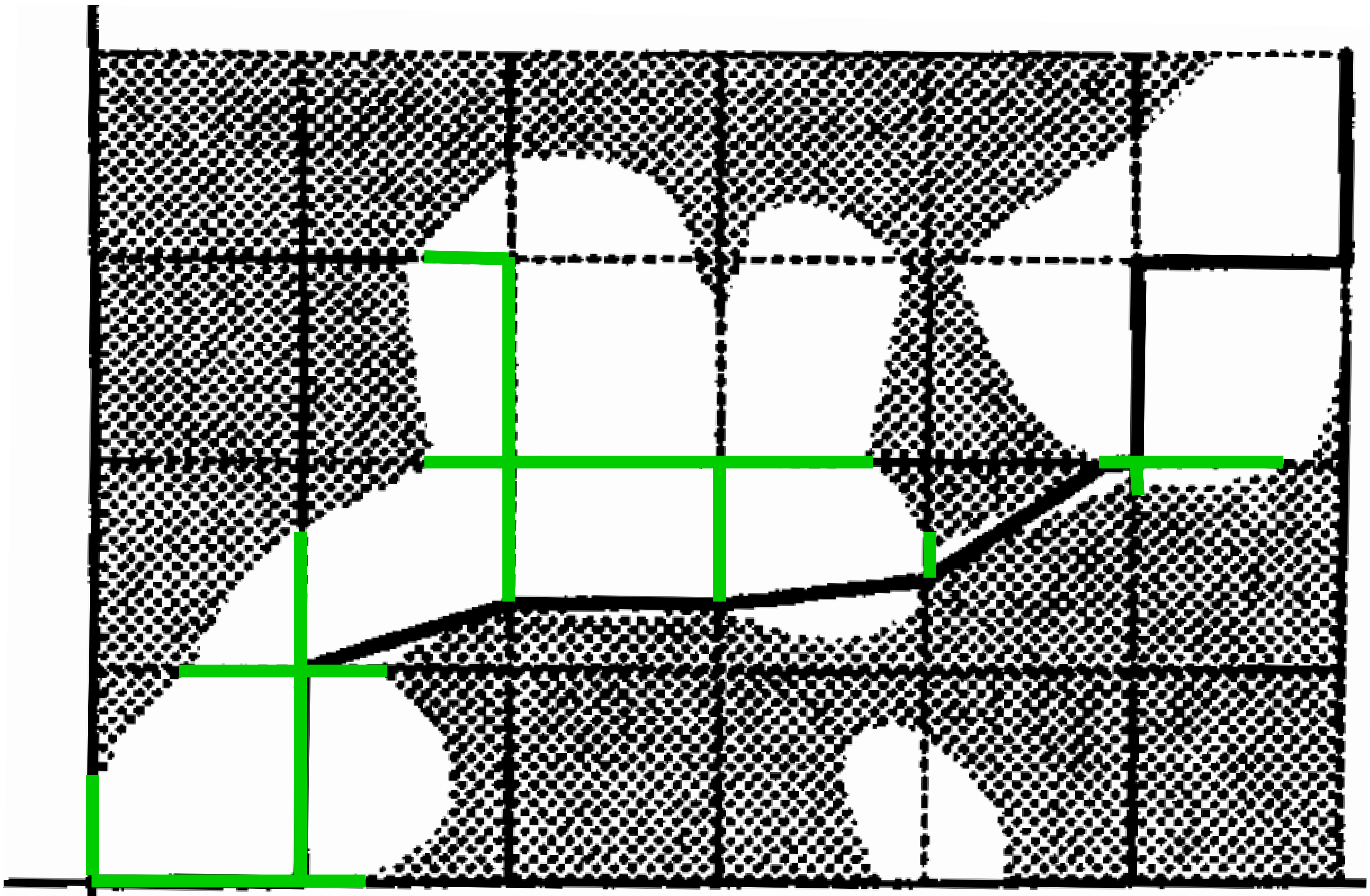
Fréchet-Distanz

Algorithmus



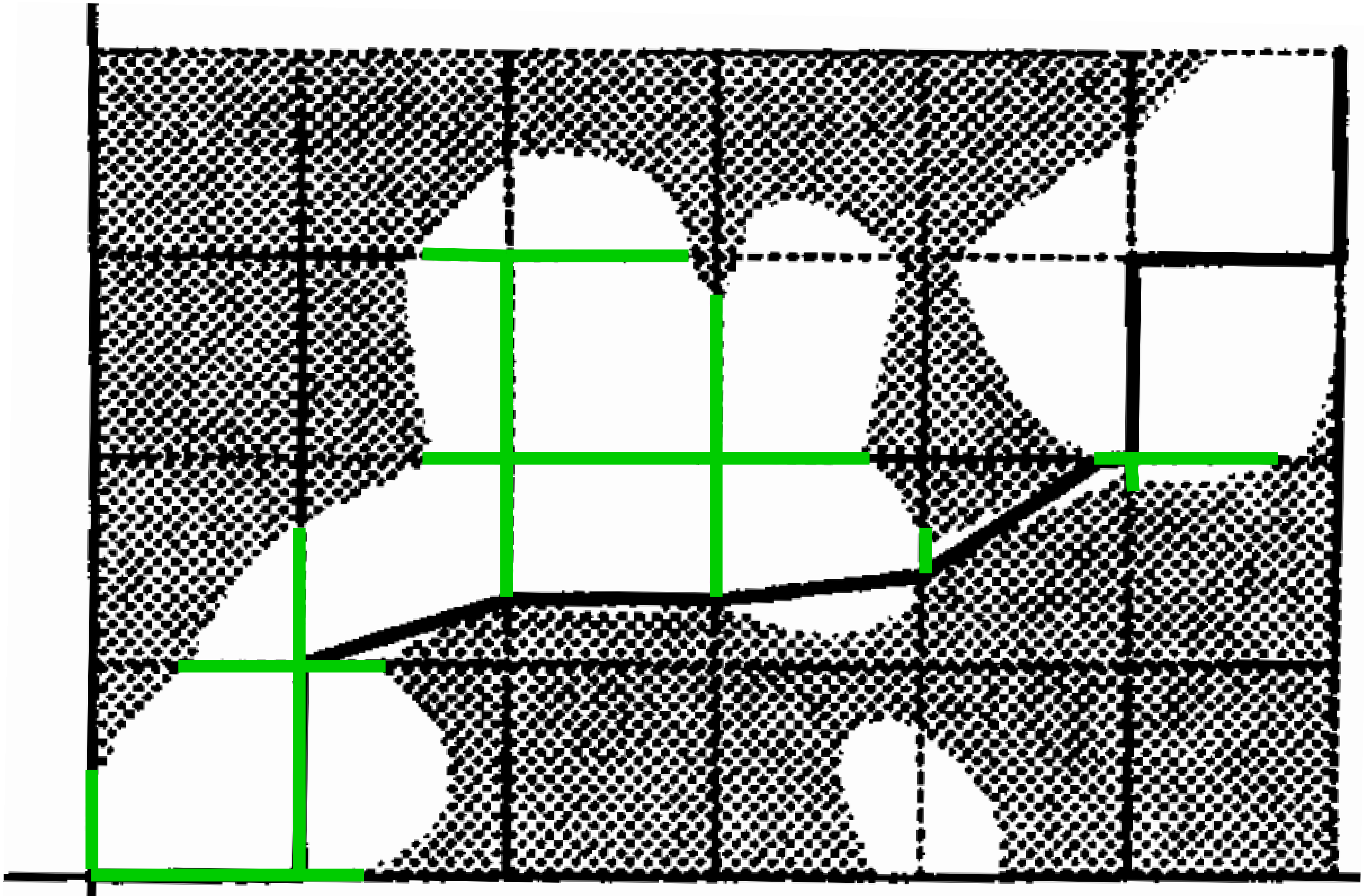
Fréchet-Distanz

Algorithmus



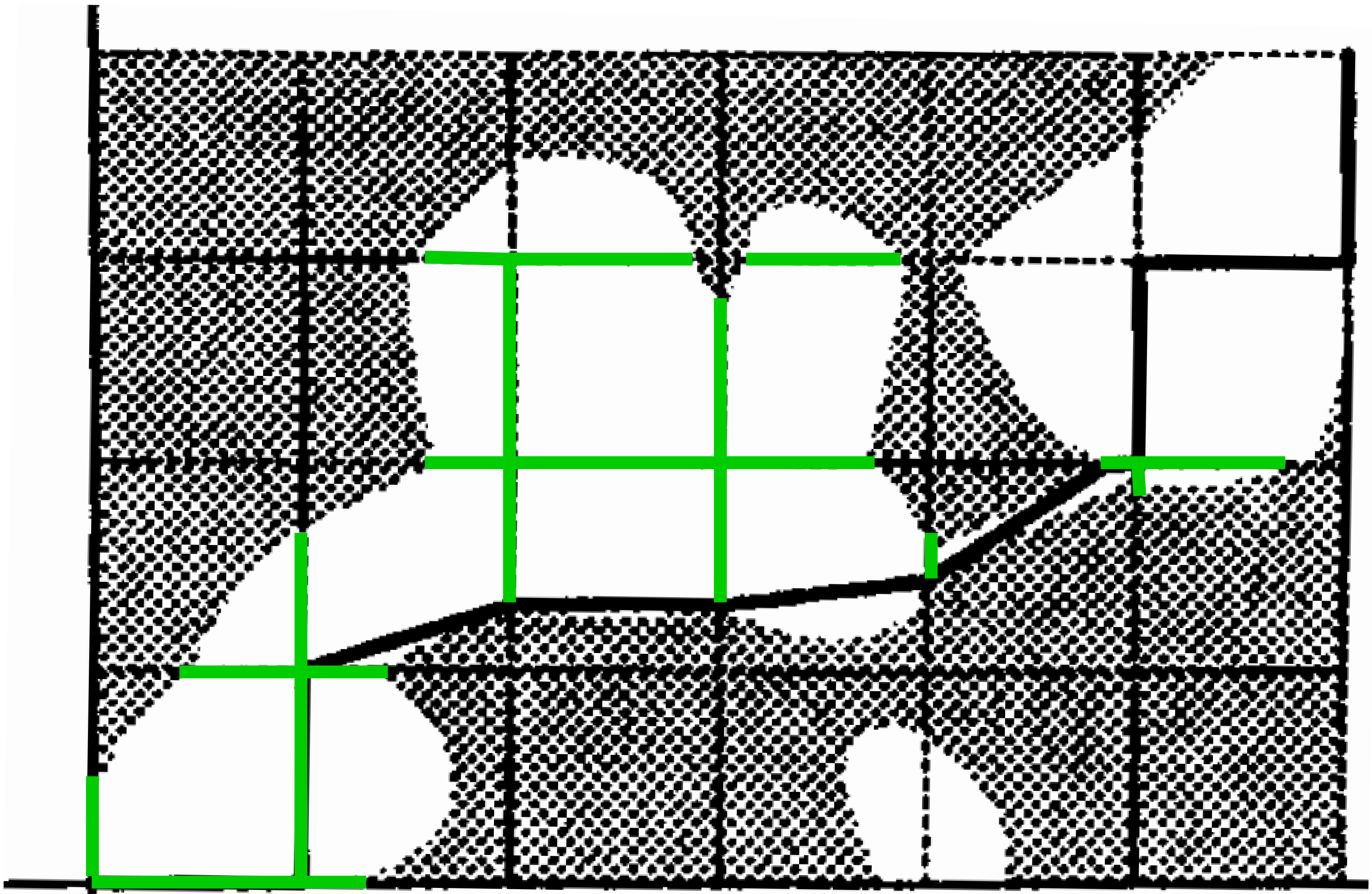
Fréchet-Distanz

Algorithmus



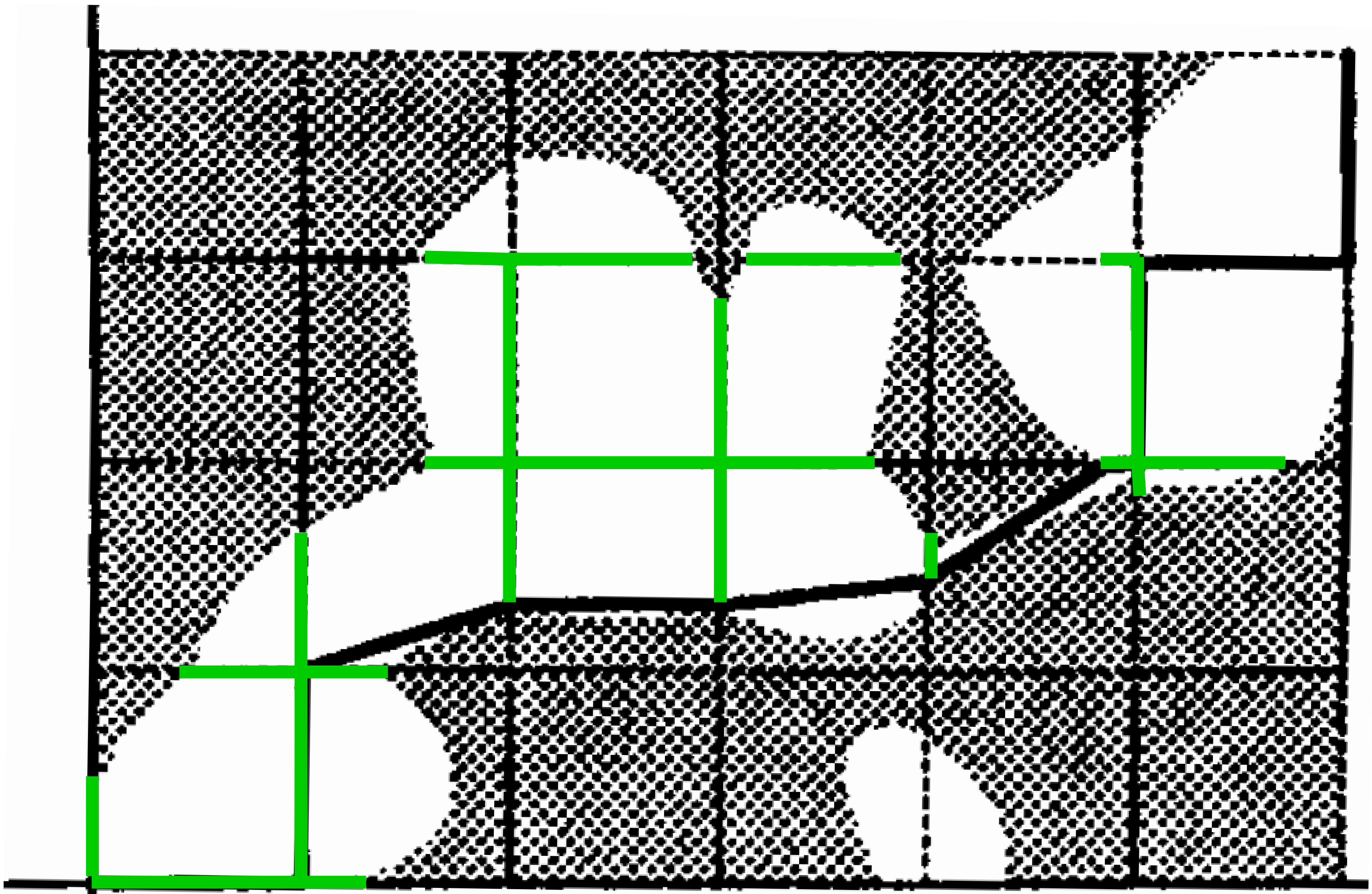
Fréchet-Distanz

Algorithmus



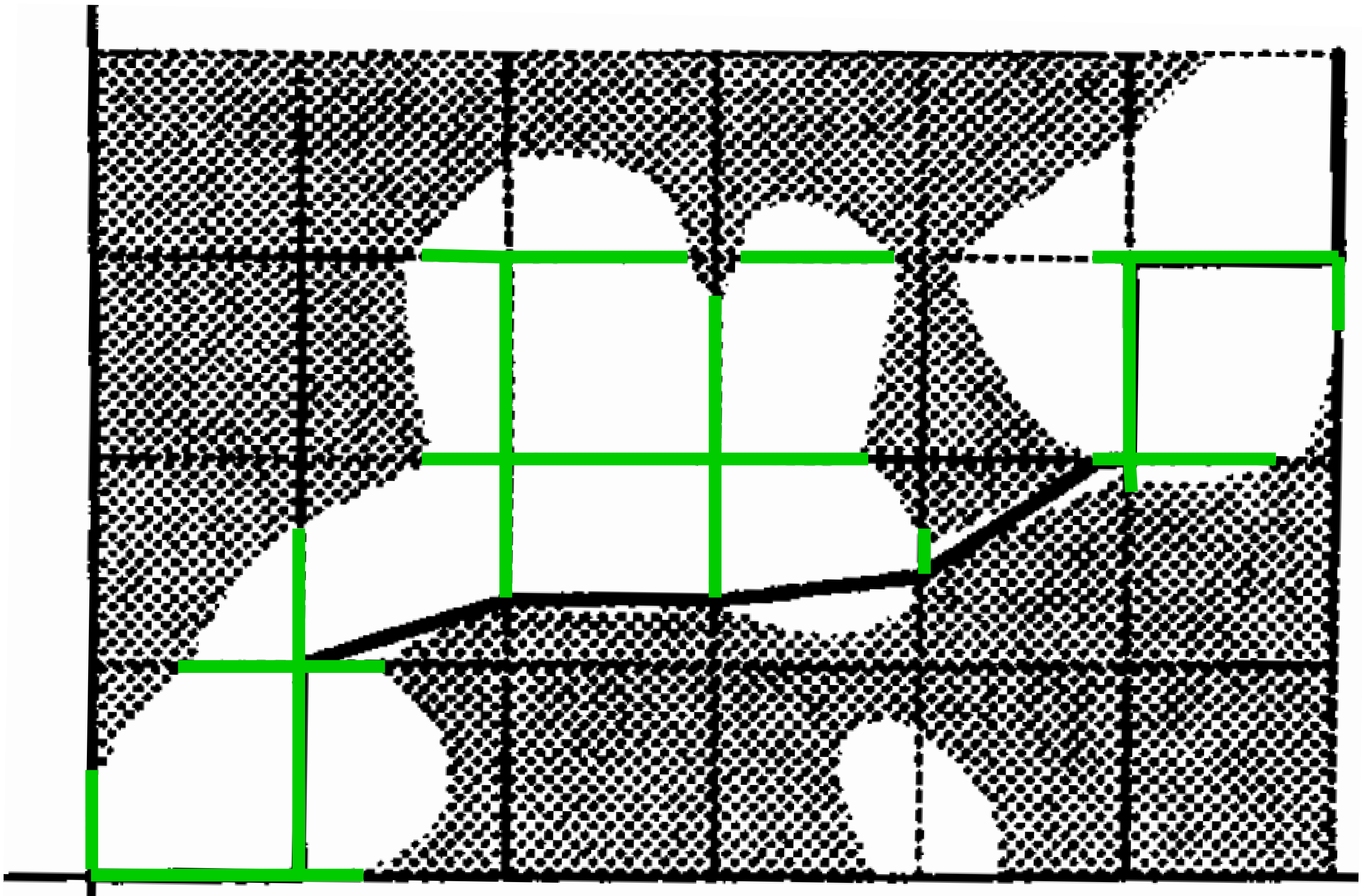
Fréchet-Distanz

Algorithmus



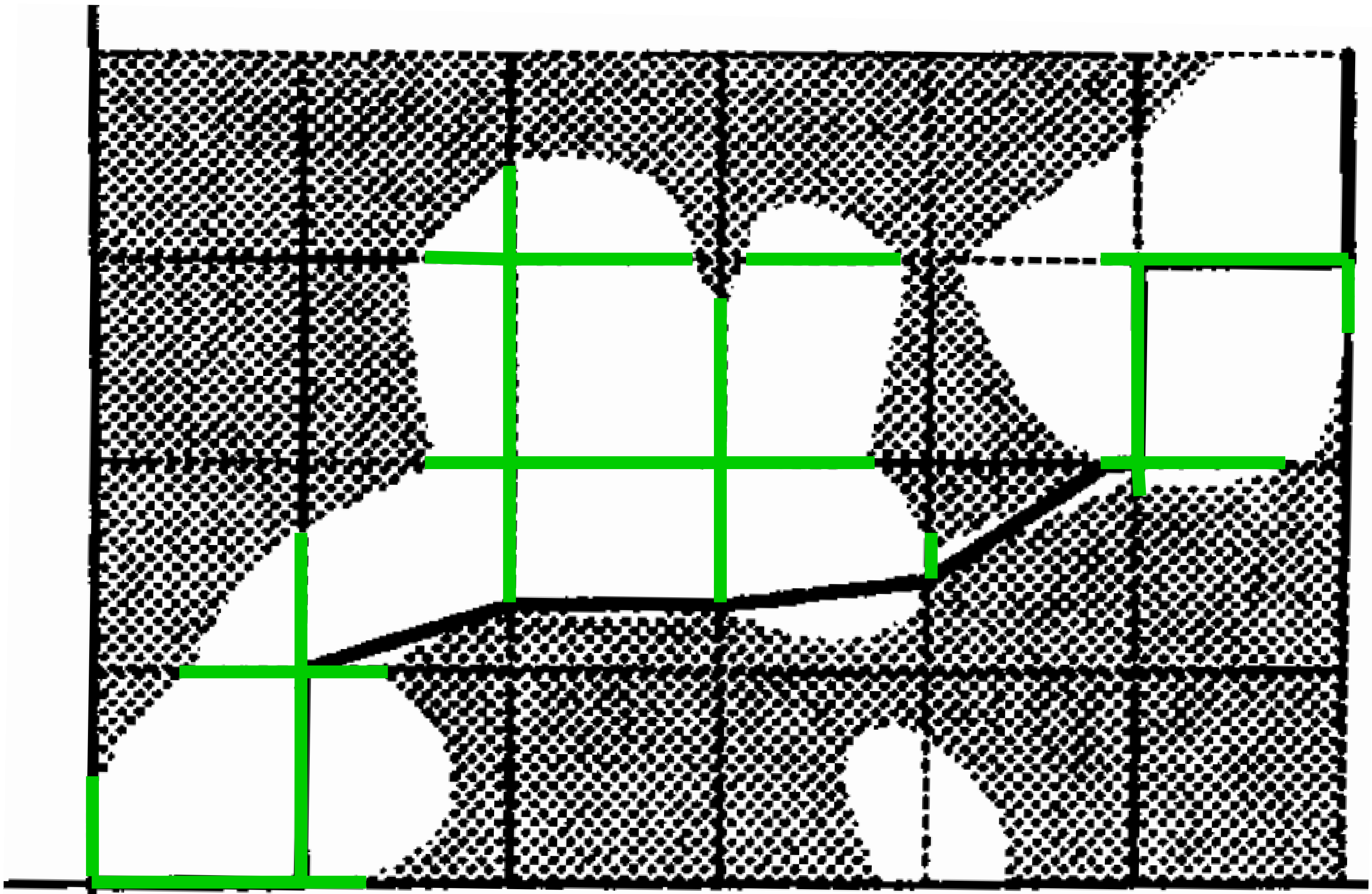
Fréchet-Distanz

Algorithmus



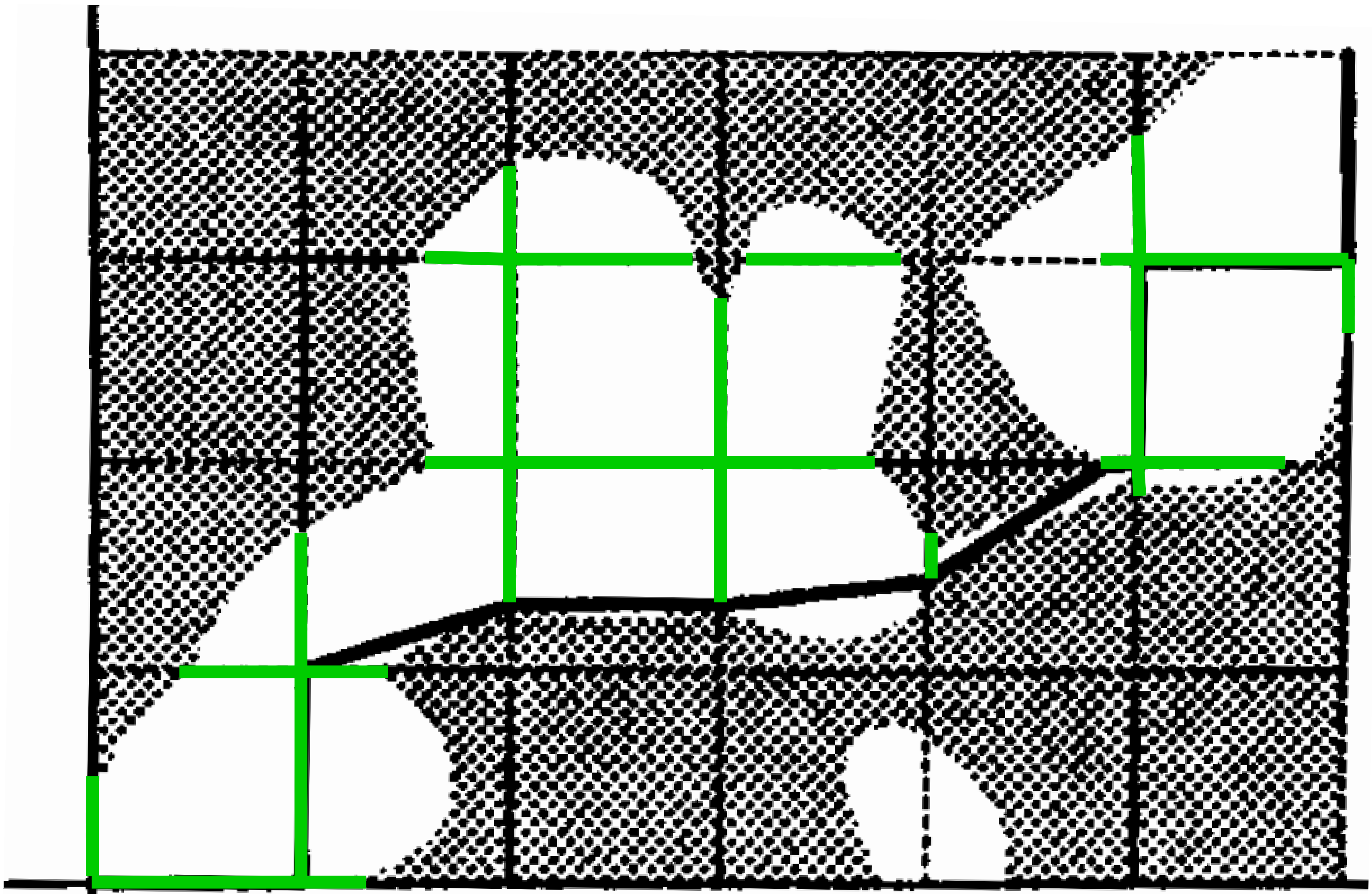
Fréchet-Distanz

Algorithmus



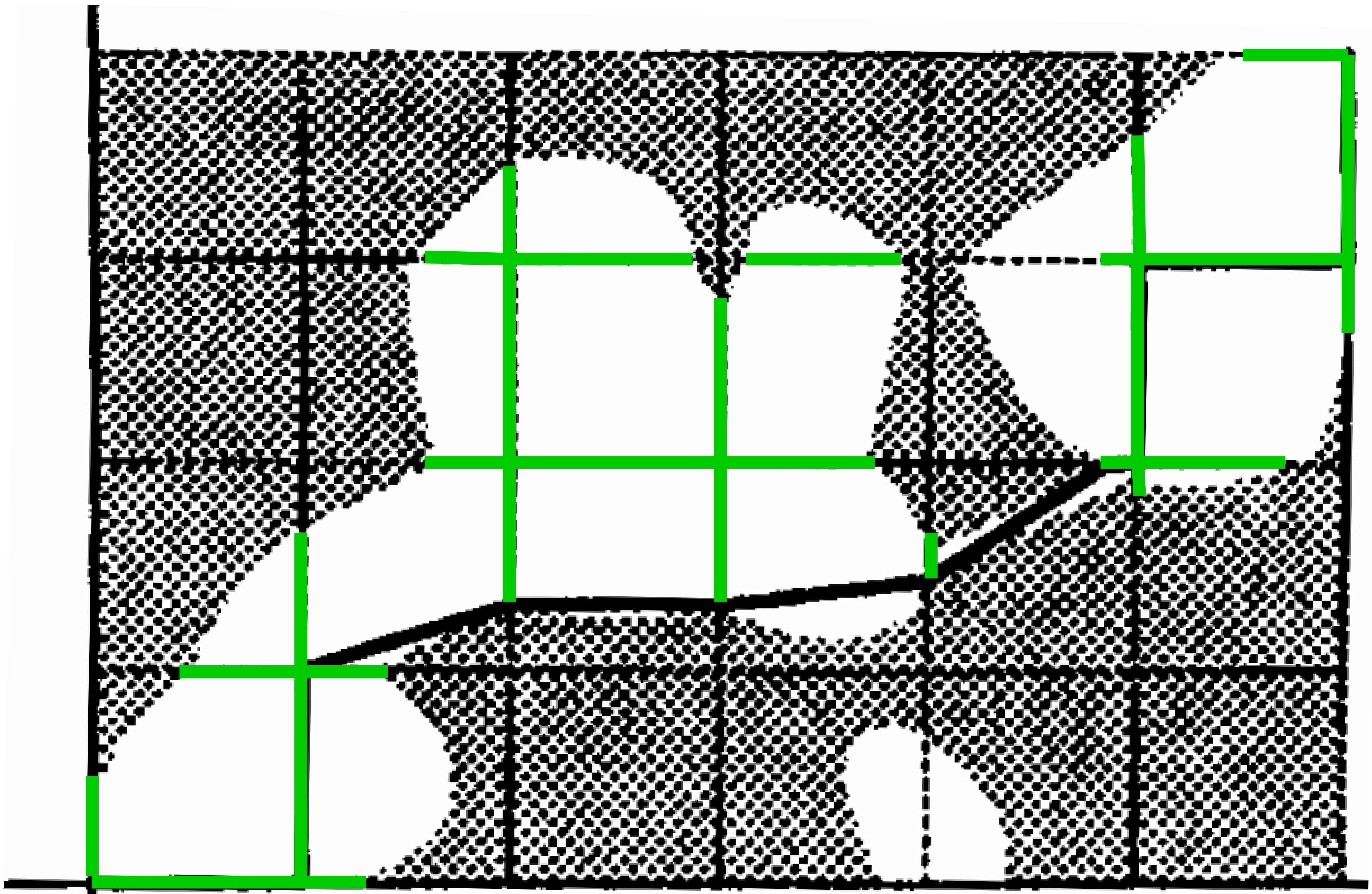
Fréchet-Distanz

Algorithmus



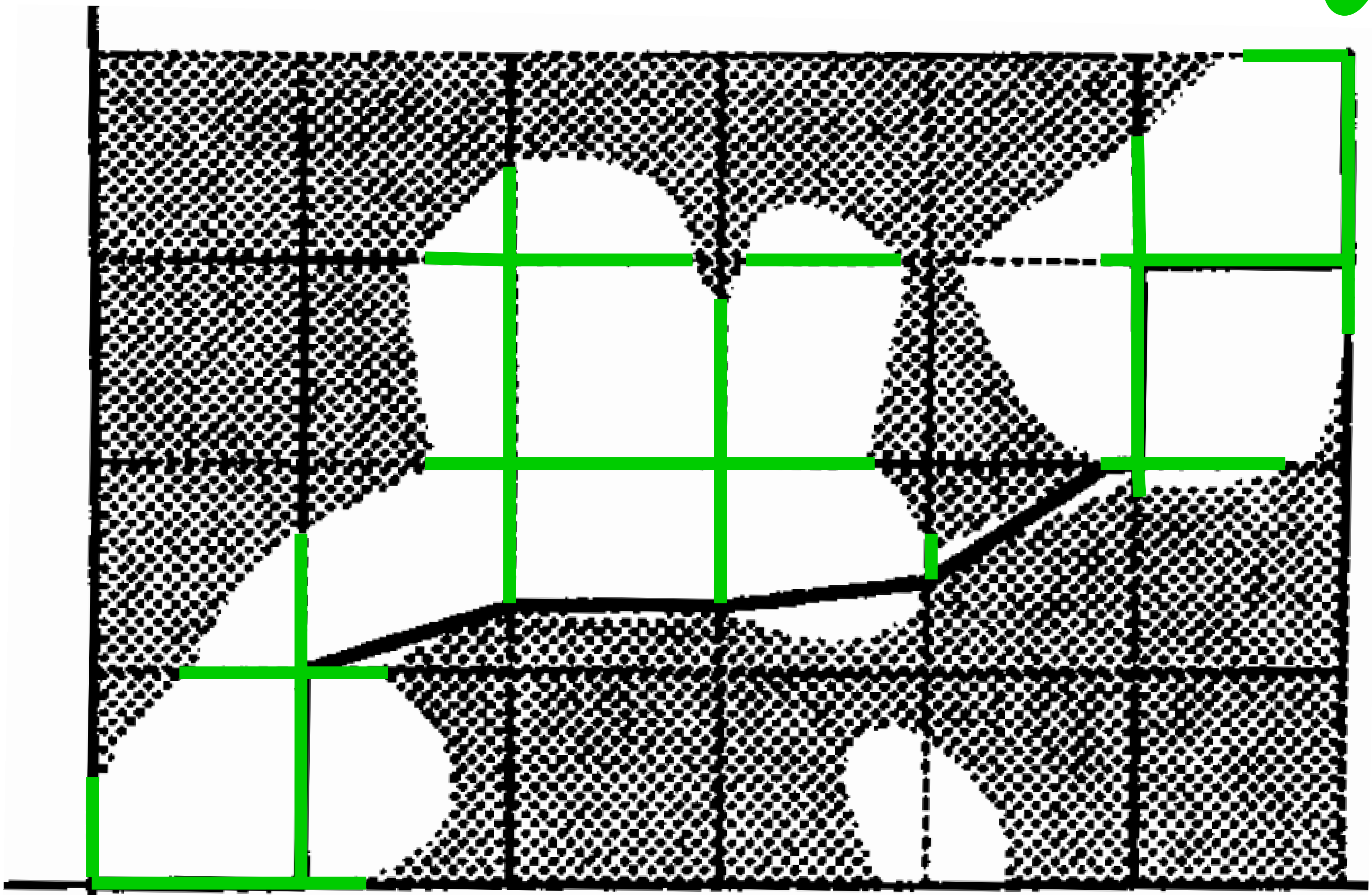
Fréchet-Distanz

Algorithmus



Fréchet-Distanz

Algorithmus

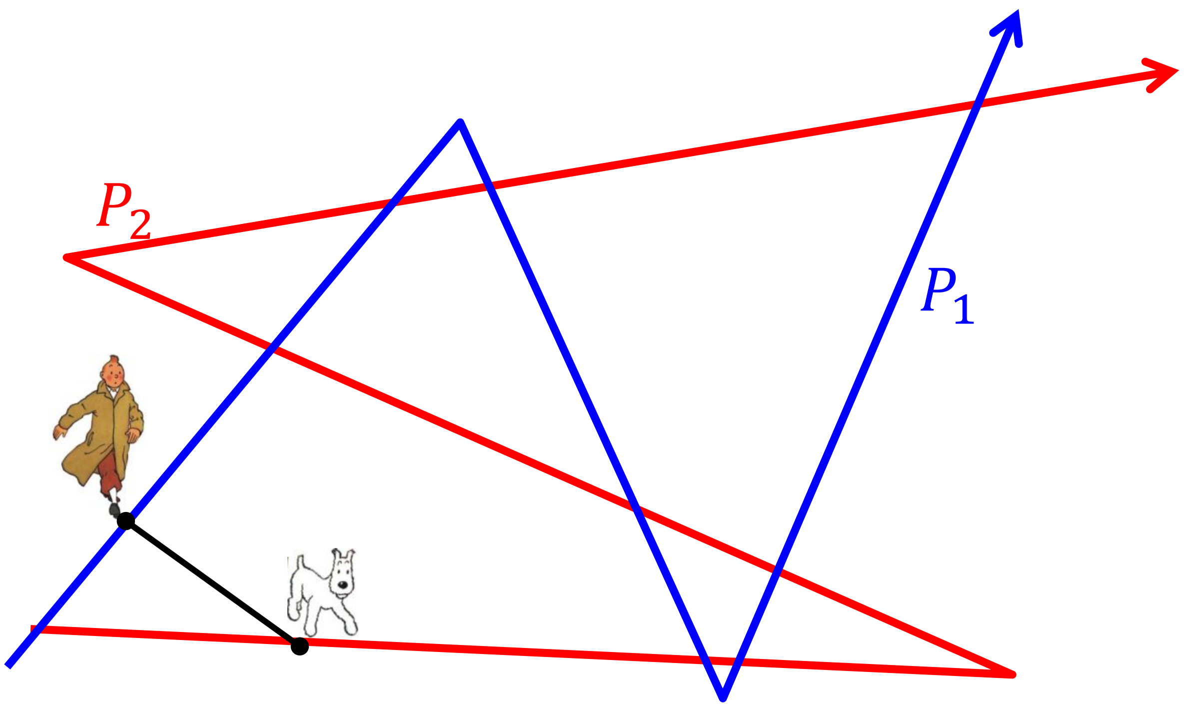


✓

Fréchet-Distanz

Berechnung:

Löse Entscheidungsproblem: Ist $d_{\text{fréchet}} \leq \epsilon$?



Fréchet-Distanz

Berechnung:

Löse Entscheidungsproblem: Ist $d_{\text{fréchet}} \leq \epsilon$?

dann parametrische Suche... $O(mn \log mn)$ Zeit

(hier ohne Details)