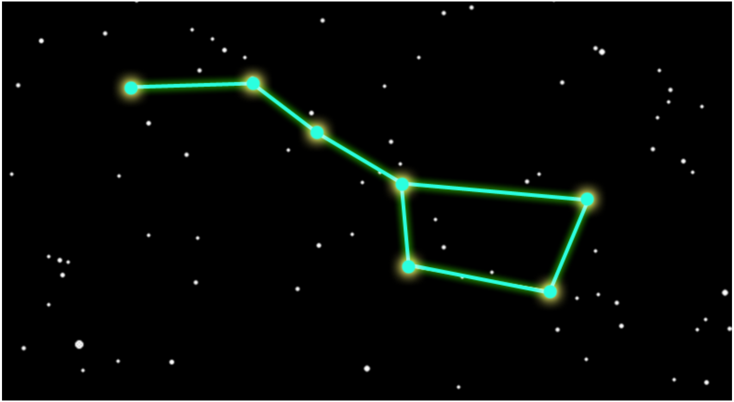
Computational Geometry

Point Pattern Matching

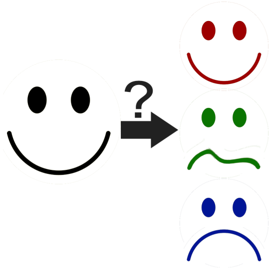
Kevin Böckler



Topics

Applications

- Distance of two shapes
  or: How similar is one shape compared to another?
- Applications:
  - Computer vision/computations
  - Molecular biology
  - Sign recognition
  - Morphing



The Problem

Input:

- two shapes (set of points), $P, Q \subset \mathbb{R}^2$
- allowed transformations T of P, Q

Output:

$f \in T$, which solves one of the following problems:

- Exact matching
- Approximated matching
- Optimal matching

Transformations

We define a transformation as follows:

Transformation

A transformation f is a function which maps one shape to another:
Let A be a shape, e.g. a set of points:
$A = \{a \in \mathbb{R}^2 | a \in A\}$
$f : \mathbb{R}^2 \rightarrow \mathbb{R}^2$
Applying f on a shape means transforming of each element by f:
$f(A) = \{f(a) | a \in A\}$

Transformations

We look at the following transformations:

Translation by t

Rotation by ϑ

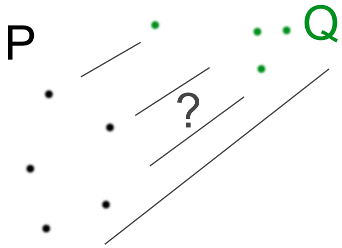
Scaling by a

Combinations

Rigid Motion means translation and rotation
Similarity means translation, rotation and scaling

Rigid Motions are interesting!

Hausdorff-Distance

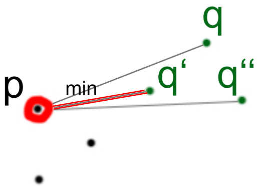


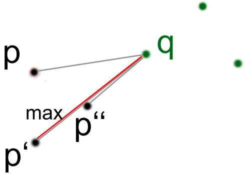
What is the distance between two shapes?
How similar are two shapes?

Hausdorff-Distance

Definition

The Hausdorff-Distance is the maximum of the minimal distances:
$\delta_H(P,Q) = \max_{p \in P} \ \min_{q \in Q} \ ||p - q||_2$





Overview of the problems

Input: $P, Q \subset \mathbb{R}^2$ and T of P, Q

Output: $f \in T$ with property of matching:

Exact matching

$\delta_H(f(P), Q) = 0$

Approximated matching

with ε as allowed error tolerance
$\delta_H(f(P), Q) \leq \varepsilon$

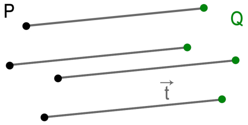
Optimal matching

$\delta_H(f(P), Q) = \min_{f' \in T} \delta_H(f'(P), Q)$

Exact matching

Definition

An exact matching of two sets P, Q is a valid function which transforms each point $p \in P$ to a point $q \in Q$.
or: $\delta_H(f(P), Q) = 0$

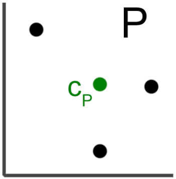


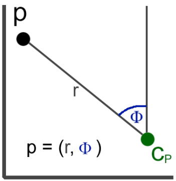
Given: sets P, Q and $T = T_{\text{translations}}$
Output: $f(x) \in T_{\text{translations}}$

Exact matching means a translated point is equal to a point of q:
$\forall p \in P : p' = f(p) = q, q \in Q$

A simple algorithm for rigid motions

1. Compute the centroids c_P, c_Q
2. Sort all points of P, Q as pairs of (Φ_i, r_i) and put them into a sequence
3. A matching is found, if the sorted sequence of P is a cyclic shift of the sequence of Q
4. If there is a matching, just compute the transformation by looking at the first pair of each P, Q





Runtime: $O(n \log(n))$

Extend with Scaling

Considering Scaling is simple:

- Start the algorithm by finding a scaling factor
  1. Compute the diameters of P, Q
  2. The scaling factor a is $\frac{d_P}{d_Q}$
- Computing scaling needs additional linear time

Exact matching does not work!

- Exact matching $\neq$ Reality
- Approximation is more effective

Problem: Approximated Matching

Approximated matching means matching all points $p \in P$ to the ε-neighborhood of a point $q \in Q$.

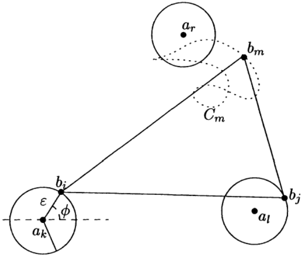
- one-to-one matching
- many-to-one matching

For a given P, Q, T, ε: is there a matching?

For a given P, Q, T: find the smallest ε

One-to-one Matching: Approaching under rigid motions

Given: sets A, B and $T = T_{\text{rigid motions}}$
Output: Is there a matching $f \in T$ so, that B is matched to A within ε?



- For each pair (b_i, b_j) find an interval of degree Φ
- All relationships are edges in a bipartite graph
- A matching does exist, if there is a perfect matching in this graph

One-to-one Matching: Approaching under rigid motions

Runtime of this approach:

1. Procedure has to be done n^4 times for all 4-tuples
2. For each procedure look at all 2-tuples' curves and calculate intervals

$\Rightarrow$ Analysis showed runtime: $O(n^8)$

Other approaches:

- Translation only in $O(n^{1.5}\log(n))$
- Disjoint ε-neighborhoods lead to $O(n^4 log(n))$
- $+$ Assuming, ε is not too close at optimal ε_{min}: $O(n^2 log(n))$

Many-to-One Matching: Hausdorff-Distance

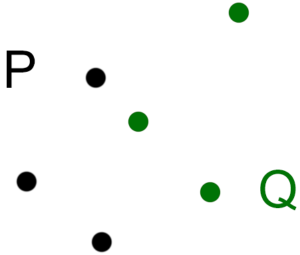
Matching by minimizing the Hausdorff-Distance

Input: Sets P, Q with $m = |P|, n = |Q|$

- Many-to-One Matching
- Hausdorff-Distance computation in
  - $O(mn)$ by computing straight forward
  - $O(m + n\log(m + n))$ by using Voronoi-Diagrams
- General idea:
  1. Take a transformation
  2. Compute the new Hausdorff-Distance
  3. Compare the result and repeat until a good transformation has been found

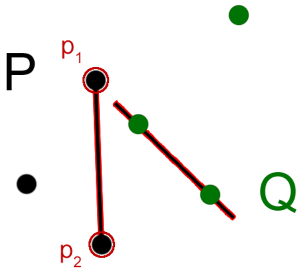
Many-to-One Matching: Goodrich approximation

1. For one diametrically opposing pair of points $p_1, p_2 \in P$:
2. Do a best match to each pair of points $q_1, q_2 \in Q$
3. Take the matching with the best resulting Hausdorff-distance $T_{\min} = \min_T \; \delta_h(T(P), Q)$



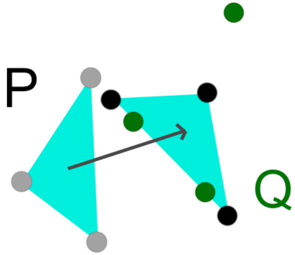
Many-to-One Matching: Goodrich approximation

1. For one diametrically opposing pair of points $p_1, p_2 \in P$:



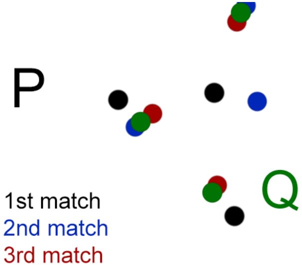
Many-to-One Matching: Goodrich approximation

1. For one diametrically opposing pair of points $p_1, p_2 \in P$:
2. Do a best match to each pair of points $q_1, q_2 \in Q$



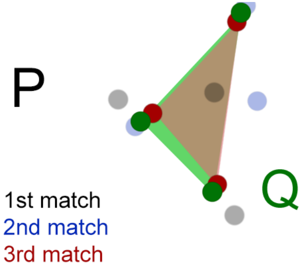
Many-to-One Matching: Goodrich approximation

1. For one diametrically opposing pair of points $p_1, p_2 \in P$:
2. Do a best match to each pair of points $q_1, q_2 \in Q$
3. Take the matching with the best resulting Hausdorff-distance $T_{\min} = \min_T \ \delta_h(T(P), Q)$



Many-to-One Matching: Goodrich approximation

1. For one diametrically opposing pair of points $p_1, p_2 \in P$:
2. Do a best match to each pair of points $q_1, q_2 \in Q$
3. Take the matching with the best resulting Hausdorff-distance $T_{\min} = \min_T \ \delta_h(T(P), Q)$



Many-to-One Matching: Goodrich approximation

1. For one diametrically opposing pair of points $p_1, p_2 \in P$:
2. Do a best match to each pair of points $q_1, q_2 \in Q$
3. Take the matching with the best resulting Hausdorff-distance

Runtime of the Goodrich approximation:

$m = |P|, n = |Q|$

1. $O(m)$
2. $O(n^2)$
3. $O(n^2 m \log(n))$

Runtime is $O(n^2 m \log(n))$

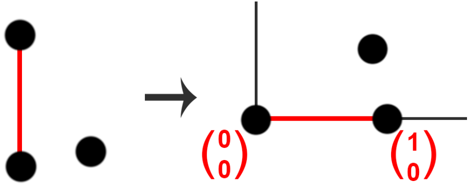
Alignment Method

Given a model M and a scene S:
Does S contain M?

Reference Frame

A reference frame of a set M in dependency of two offset points $a, b \in M$ represents a new coordinate system with properties:

- a is assigned as the origin $(0, 0)$
- b is assigned as an alignment vector $(1, 0)$
- all other points relate to these two points



Alignment Method

Given a model M and a scene S:
Does S contain M?

The Algorithm:

1. Create reference frames for each pair $a, b \in M$
2. Create reference frames for each pair $p, q \in S$
3. Find one reference frame of M, whose points lie all in a ε-neighborhood of a reference frame of S

Alignment Method

- Provides Similarity-Transformation
- Exhaustive method
- Worst-case runtime is $O(|M|^3|S|^2\log(|S|))$

Question: Given multiple scenes S_i, does one of them contain M?
or: Given multiple models M_i, does S contain a model M_i?

- Solution: Hashing of models and scenes
- Voting in hashtables

Short Summary: Matching of Point Patterns

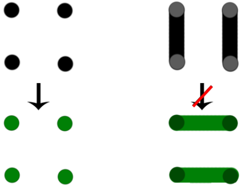
Algorithms:

- Exact matching: Sorting and searching for vectors
- Approx matching: Theoretical argumentation
- With respect to the Hausdorff-distance: Goodrich Approximation
- Pattern recognition: Alignment Method

Outlook:

- 3D with projections
- Randomization? - Not yet

The difference of line segments and point sets



New Problem

Given two sets of line segments A, B with cardinalities $n = |A|, m = |B|$.

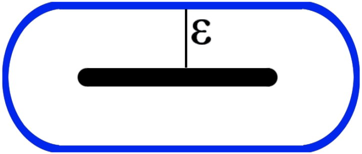
- infinite points
- Hausdorff-Distance:
  - harder to compute
  - with voronoi diagram intersections: $O(n \log(n))$

Translations for approximated matching

Definition

A **racetrack** A_ε is a disk of radius ε around a given line segment $A_\varepsilon = A \oplus C_\varepsilon$, where C_ε is a circle with radius $\frac{\varepsilon}{2}$

Example: set A, consisting of a single line seqment

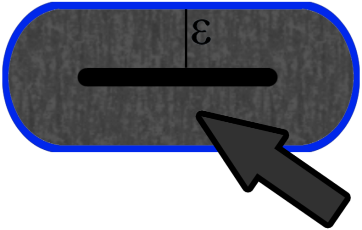


Translations for approximated matching

Definition

A **racetrack** A_ε is a disk of radius ε around a given line segment $A_\varepsilon = A \oplus C_\varepsilon$, where C_ε is a circle with radius $\frac{\varepsilon}{2}$

Example: set A, consisting of a single line seqment

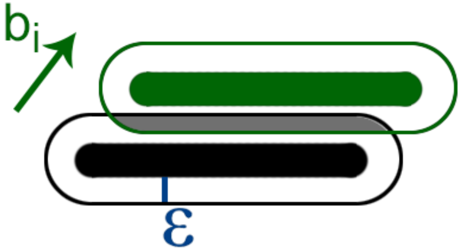


Translations for approximated matching

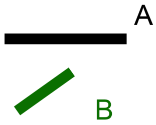
A translated racetrack

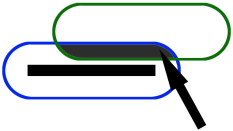
Let A_i^ε be the intersection of A_ε and the translated racetrack A_ε by a vector b_i

$A_i^\varepsilon = A_\varepsilon \cap (A_\varepsilon \oplus b_i)$



Translations for approximated matching





Theorem

There is a matching exactly if $S = \bigcap_{i=1}^{m} A_i^{\varepsilon} \neq \emptyset$.

This is aquivalent to the existence of a cell with depth m

- Depth determination with line sweep algorithm
- Complexity: mn arcs and lines, $(mn)^2$ intersections points

$\Rightarrow$ Translations are performed in $O((mn)^2 \log(mn))$

Extend with Rotations

Procedure:

1. Rotate arrangement around the origin by $\theta \in (0, 2\pi)$
2. Translate as before

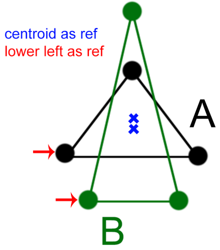
Difficulty: Arrangement changes with rotation

- Looking at events, when the arrangement changes
- Rigid motions are performed in $O(m^3 n^3 \log(mn))$

Idea of reference points

Definition

A reference point r_A is a representive point of a shape A so that a perfect matching of A to B bounds the distance of $T(r_A)$ to r_B by a constant factor a, which is called the *quality of the reference point*
$a\ \delta(T_{\text{perfect}}(A), B) = \delta(T_{\text{perfect}}(r_A), r_B)$



Which reference point has the higher local quality?

Reference point for translation

Theorem

Translating A to B by matching r_A to r_B is at most $a+1$ times worse than the optimal matching

- Finding a reference point as above needs linear time
- Translating r_A to r_B needs constant time

The translation algorithm has a runtime of $O(n)$

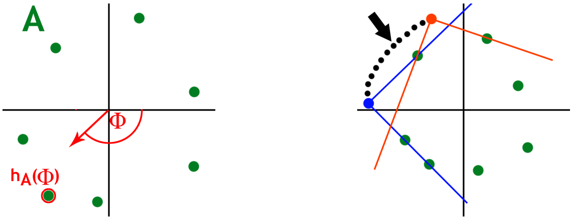
Steiner point for rigid motions

What about rigid motions?

- Center of the boundary of the convex hull: $a = 4\pi + 4 \approx 16.6$
- Steiner point:
  - Input has to be a convex body
  - Works for similarities
  - $a = \frac{4}{\pi} \approx 1.27$ (2D)
  - $a \approx 1.5$ (3D)

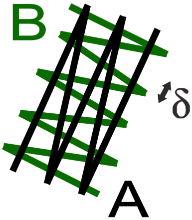
Steiner point for rigid motions

1. Rotate lower-left Reference Point around A
2. Keep orthogonal tangents to A
3. Take the average over all those rotated points



- Support function $h_A(\Phi)$ is largest extent in direction of Φ
- The desired point is $h_A(\Phi)\begin{pmatrix}\cos\Phi\\ \sin\Phi\end{pmatrix}$
- Average is found by $\frac{1}{\pi}\int\limits_0^{2\pi} h_A(\Phi)\begin{pmatrix}\cos\Phi\\ \sin\Phi\end{pmatrix} \mathrm{d}\Phi$

Distance for line segments



Problem:
Two sets A, B with quite small Hausdorff-distance δ_H but high disparity

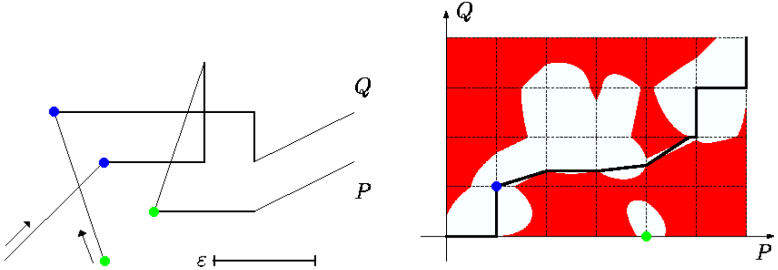
Better description of distance

The Fréchet Distance is the greatest distance which can appear when walking along two monoton paths through both given shapes

Fréchet Distance

Parametrization with a P/Q - diagram

- Input: Polygonal chains of line segments P, Q and distance ε
- Output: *True*, if $\delta_F(P,Q) \leq \varepsilon$



- P/Q - diagram needs runtime $O(mn)$ with $m = |P|, n = |Q|$

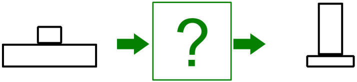
Short Summary : Matching of curves & areas

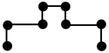
- Shapes are more complex then sets of points
- Distances of curves are different from point patterns

Methods:

- Racetrack intersections
- Reference Point approximation
- Fréchet Distance

Introduction



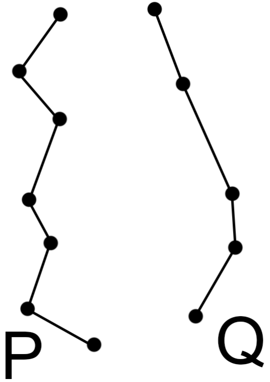


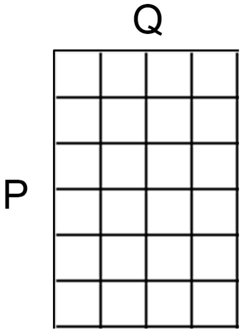
Definition

Let P and Q be two shapes of size $m = |P|, n = |S|$. Morphing describes the mapping of P to Q.

- The general process of morphing:
  1. Find features and match them
  2. Motions for each feature pair
  3. Transformation $=$ combining motions $+$ constraints
- For simplicity we will look on polygonal chains

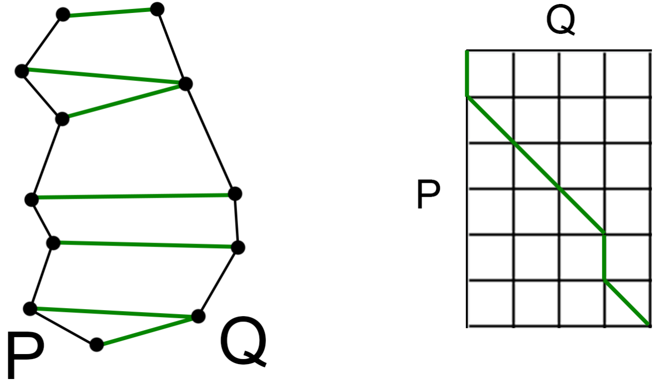
Linear chain matching





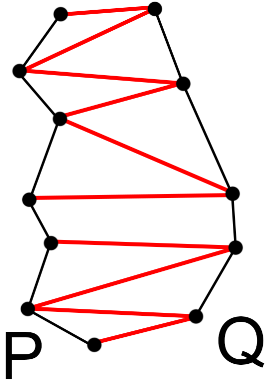
1. Construct a grid of $P \times Q$
2. Monotone walkthrough the grid represents matching

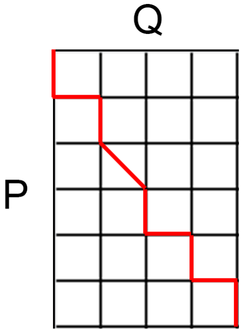
Linear chain matching



1. Construct a grid of $P \times Q$
2. Monotone walkthrough the grid represents matching

Linear chain matching





1. Construct a grid of $P \times Q$
2. Monotone walkthrough the grid represents matching

Linear chain matching

Optimal path in $O(nm)$ with dynamic programming

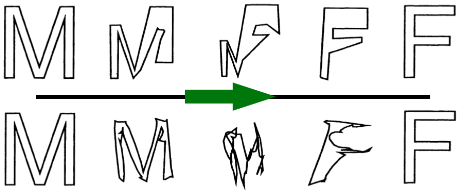
What are suitable matching constraints?

- Shortest distance
- parallel matching line segments

Last step:

- For each matching $(p, q) \in P \times Q$ do a linear mapping

Difficulties with chain matching



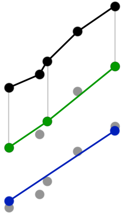
- Goal: minimize intersections
- Reason: Choosing the wrong origin from the shape
- Solution: Select constraints
  - e.g. keep parallel sides while morphing
  - runtime: $O(n \log(n))$

Lessons learned

- Definition, features and similarities of shapes
  - elements of a shape
  - distance
  - transformations
- Approximative approaches matching two sets of points
  - Goodrich approximation
- Finding a shape in another one
  - Alignment method
- Areas are point sets with infinite elements
  - Complex shapes
  - Reference Points

Outlook:

- Three dimensions
- Shape simplification
  - Redundancy
  - Complexity



Sources

Alt, H., Guibas, L. (1999) Discrete Geometric Shapes: Matching, Interpolation, and Approximation given in Sack, J., Urrutia, J. (1999) *HANDBOOK OF COMPUTATIONAL GEOMETRY*

Goodrich, M., Mitchell, J., Orletsky, M. (1999) *Approximate Geometric Pattern Matching under Rigid Motions*

Pelletier, S. (2002) *Computing the Fréchet distance between two polygonal curves*

Alt, H., Aichholzer, O., Rote, G. (1994) *Matching Shapes with a Reference Point*