Transport on shape manifold : matching close shapes

Guillaume Charpiat

September 2008

Introduction

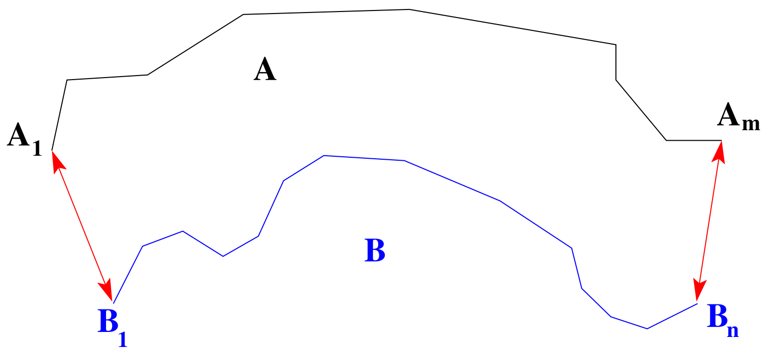
The key idea to fill in the holes in a sample set of silhouettes (whose intrinsic dimension is far too high to expect at anytime a dense sample) consists in computing deformation fields between close shapes (since between farther shapes it won't be reliable at all) and expressing statistics on them. But one cannot compare two deformations which are defined on two different shapes, so we have to *transport* deformation fields from shapes to shapes. This transport will be based on... deformations between close shapes. So now we have two good reasons to search for ways to match two close shapes together.

This document is organized as follow : first, explain the basic tool used (and its first extension to ensure coherency) on a simple example (two open curves to be matched). Then, extension to matching arbitrary shapes (set of closed connected components). This matching algorithm is not perfect, but we don't need a perfect one (later, we will *learn* the metric and the ways to match shapes, here it's just a first step to extract some information from a set of shapes). We could consider any other way to match close shapes, there are plenty of them, we just need any which deals with non-trivial topology (a shape may have several connected components, or holes), which deals with matching two shapes with different topologies (it happens very often), and which is not too slow and does not give too bad matchings, of course. Finally, we define and compute transport based on the matching tool.

Contents

1 Matching close shapes

1.1 Simple case of two open curves sharing starting point : *Dynamic Time Warping*



Let A and B be two polygons, with m and n vertices respectively. We know that A_1 should be matched with B_1, and A_m with B_n. The question is : what is the best matching function f, from $[|1, m|]$ to $[|1, n|]$, so that A_i correspond to $B_{f(i)}$? Here all vertices A_i will be matched to vertices of B (and not to intermediate points on segments). Consequently it may happen that several successive A_i be mapped to the same point of B or that a vertex B_j has no corresponding point on A.

Requirements:

- $f(1) = 1$
- f is an increasing function, i.e. $\forall i,\ f(i+1) \geqslant f(i)$
- $f(m) = n$

We suppose that A and B have already been rigidly registered (i.e. with respect to position, size, and possibly orientation if it makes sense). Consider the simple energy to minimize, subject to previous conditions :

$$E(f) = \sum_i \|A_i - B_{f(i)}\|^2.$$

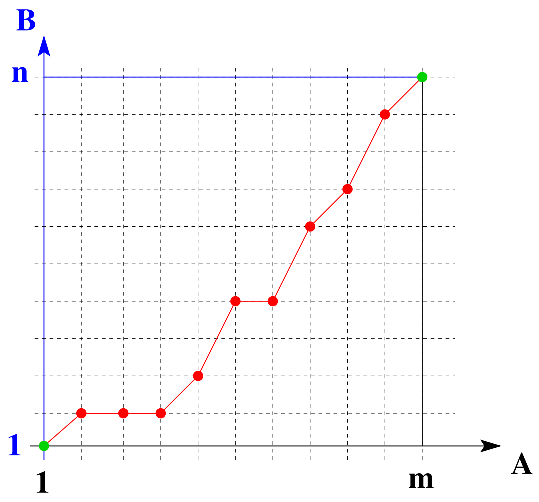


Figure 1: The graph of a matching function f, i.e. here any increasing function, with fixed extremities.

Drawing the graph of f is interesting (see figure 1) and inspires the *dynamical time warping* algorithm:

- Start with $f(1) = 1$. Cost is fixed : $\|A_1 - B_1\|$.
- Check all possible values of $f(2)$, i.e. between 1 and n and their associated costs of the matching so far : $\|A_1 - B_1\| + \|A_2 - B_{f(2)}\|$ (and store these values).

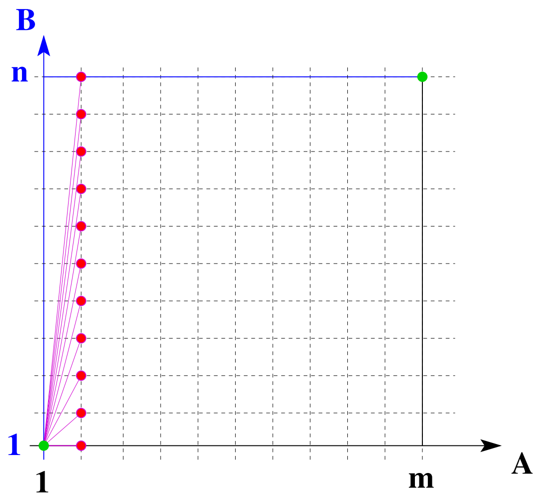


Figure 2: Compute the cost of all possible matchings for A_1 and A_2.

- Check all possible values of $f(3)$, i.e. between 1 and n and their associated costs (for the best possible path ending in $(A_3, B_{f(3)})$):

$$\inf_{1 \leqslant i_2 \leqslant f(3)} \Big(\|A_1 - B_1\| + \|A_2 - B_{i_2}\| \Big) \ + \ \|A_3 - B_{f(3)}\|.$$

  For each possible value of $f(3)$, store the cost and the associated best path $(1, i_2, f(3))$.

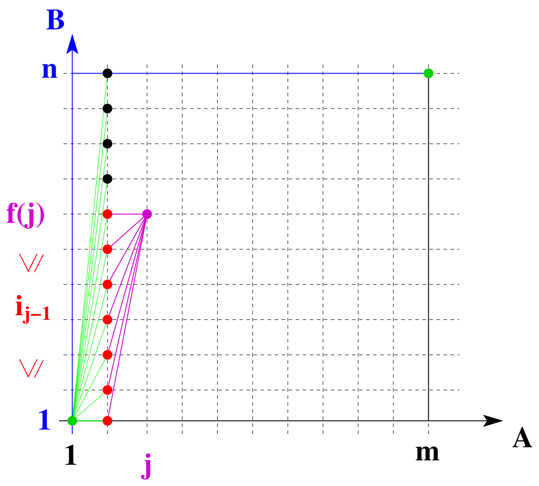


Figure 3: Step j $(=3)$. For all possible values of $f(j)$, check all possible sequences of matchings for $(A_1, A_2, \ldots, A_j)$ (with cost $C_j(f(j)) = C_{j-1}(i_{j-1}) + \|A_j - B_{f(j)}\|$), relying on all possible best sequences of matching for $(A_1, A_2, \ldots, A_{j-1})$.

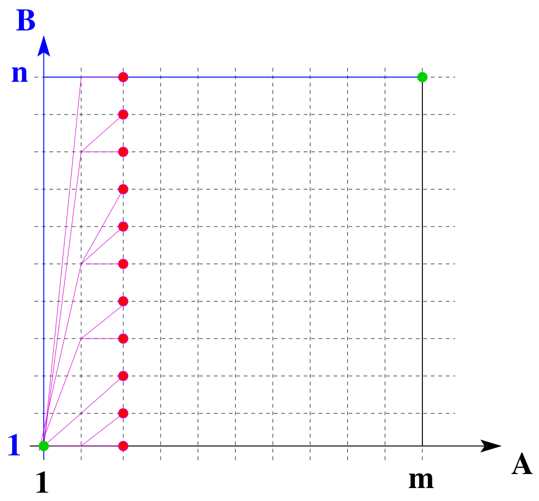


Figure 4: All best sequences of matchings up to A_j.

- Iteratively, for each j, we already know all best paths from (A_1, B_1) up to all possible pairs (A_{j-1}, B_k) at previous point, and we check all possible values of $f(j)$ and their associated costs (for the best possible path ending in $(A_j, B_{f(j)})$):

$$\begin{aligned}
C_j(f(j)) &= \inf_{1\leqslant i_2\leqslant i_3\leqslant\cdots\leqslant i_{j-1}\leqslant f(j)} \left(\sum_{1\leqslant k<j} \|A_k - B_{i_k}\| \right) + \|A_j - B_{f(j)}\| \\
&= \inf_{1\leqslant i_{j-1}\leqslant f(j)} \left(\inf_{1\leqslant i_2\leqslant i_3\leqslant\cdots\leqslant i_{j-1}} \sum_{1\leqslant k<j-1} \|A_k - B_{i_k}\| + \|A_{j-1} - B_{i_{j-1}}\| \right) \\
&\qquad + \|A_j - B_{f(j)}\| \\
&= \inf_{1\leqslant i_{j-1}\leqslant f(j)} C_{j-1}(i_{j-1}) + \|A_j - B_{f(j)}\|
\end{aligned}$$

and store the cost and the associated best path $(1, i_2, i_3, \ldots, i_{j-1}, f(j))$.

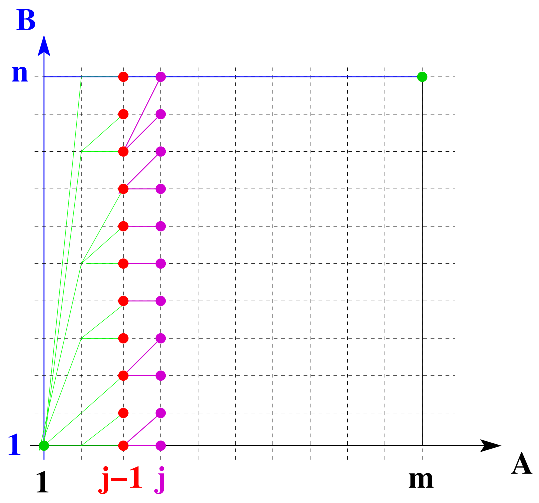


Figure 5: Best paths up to A_j, based on best paths up to A_{j-1}.

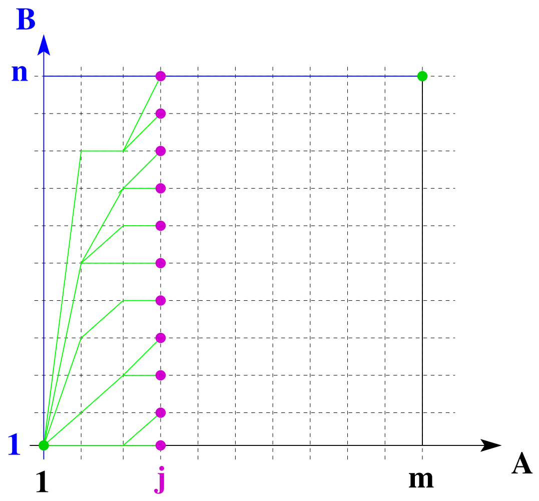


Figure 6: The same. Iterate.

At the end of the process: the best match between A and B is the sequence $(i_1 = 1 \leqslant i_2 \leqslant i_3 \leqslant \cdots \leqslant i_{m-1} \leqslant i_m = n)$ that minimizes $\sum_k \|A_k - B(i_k)\|$.

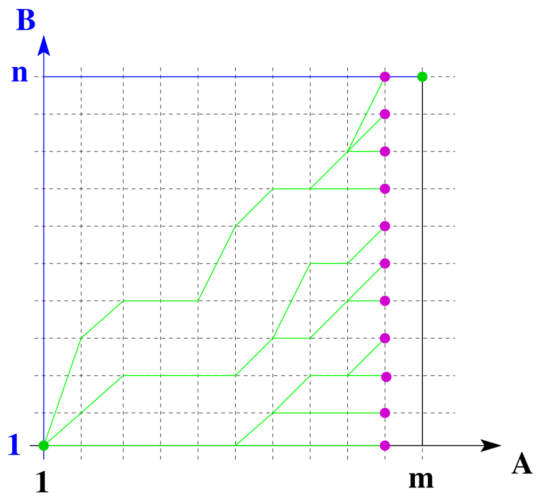


Figure 7: Best paths up to A_{m-1}, with associated costs $C_{m-1}(f(m-1)$.

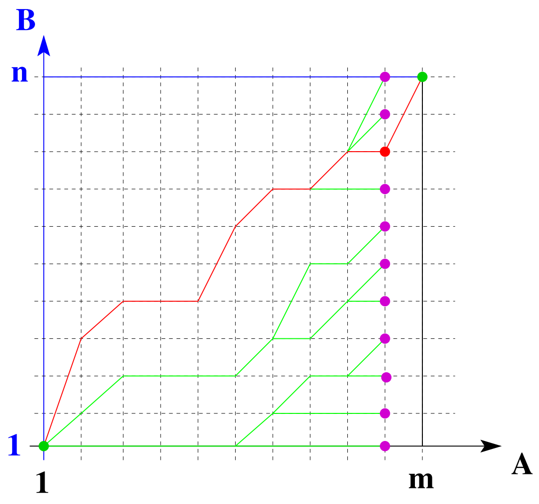


Figure 8: Building best path to A_m.

$$\begin{aligned} \inf_{i_1=1\leqslant i_2\leqslant i_3\leqslant\cdots\leqslant i_{m-1}\leqslant i_m=n} \left(\sum_k \|A_k - B_{i_k}\|\right) &= C_m(n) \\ &= \inf_{1\leqslant i_{m-1}\leqslant i_m=n} C_{m-1}(i_{m-1}) \;+\; \|A_m - B_n\| \end{aligned}$$

Since at each step we have successively computed stored the best paths, the problem is solved and we have the global solution.

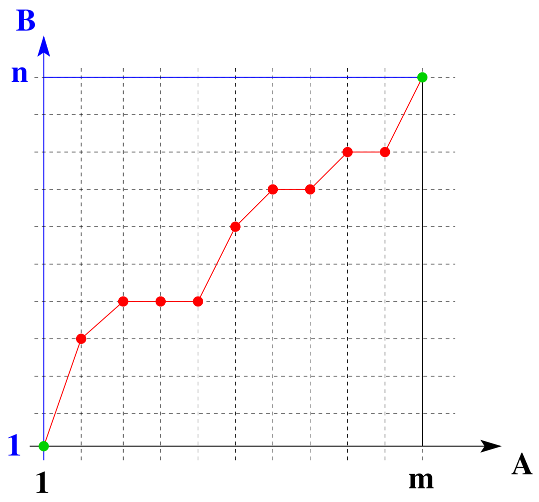


Figure 9: Final path is the optimal matching.

Complexity :

- Number of steps : number of points on A (m)
- For each step j : try all matching possibilities for A_j (number of points on B: n)
- For each possible matching $f(j)$ for A_j : try to connect to all possible best paths ending in (A_{j-1}, B_k) with $k \leqslant f(j)$.

The third loop can be achieved at the level of the second one ($f(1) = 1$, $f(2) = \max(C_{j-1}(1), C_{j-1}(2))$, $f(3) = \max(C_{j-1}(1), C_{j-1}(2), C_{j-1}(3)) = \max(f(2), C_{j-1}(3))$, etc. : complexity $= n$ for all $f(j)$ together).

Total: $m \times n$ (instead of n^m for trying successively all possible matchings)

1.2 Variations on *Dynamic Time Warping*

1.2.1 Search for more coherency

Let us replace the energy to minimize $E(f) = \sum_i \|A_i - B_{f(i)}\|^2$ by:

$$E(f) = \sum_i \left(\|A_i - B_{f(i)}\|^2 \; + \; \beta \, \|\overrightarrow{A_i A_{i+1}} - \overrightarrow{B_{f(i)} B_{f(i+1)}}\|^2 \right) \tag{1}$$

(where β is any positive constant) in order to enforce the spatial similarity which happens when following the contours A and B. Note that :

$$\|\overrightarrow{A_i A_{i+1}} - \overrightarrow{B_{f(i)} B_{f(i+1)}}\|^2 = \|\overrightarrow{A_i B_{f(i)}} - \overrightarrow{A_{i+1} B_{f(i+1)}}\|^2$$

i.e. the norm of the difference between the red vectors is the same as the one between the green vectors on figure 10. We want this color vectors to be as identical as possible.

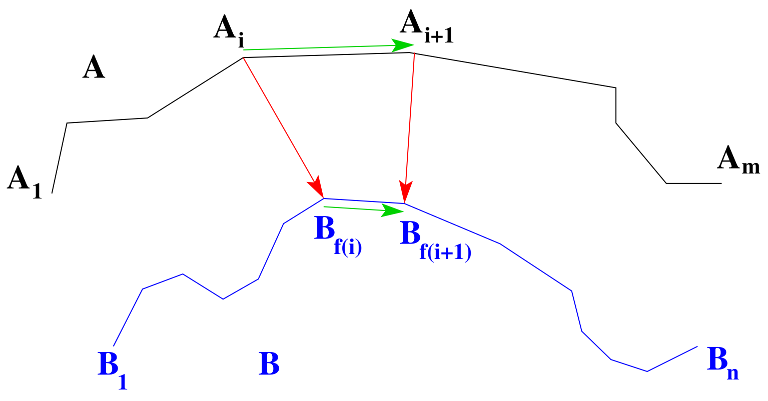


Figure 10: Enforcing spatial coherency : by minimizing the difference between the red vectors or between the green ones.

Denoting $\overrightarrow{A_i B_{f(i)}}$ by $\overrightarrow{v_i}$, the energy can be rewritten as follow:

$$E(f) \;=\; \sum_i \; \overrightarrow{v_i}^2 \;+\; \beta \, \|\overrightarrow{v_i} - \overrightarrow{v_{i+1}}\|^2 \;=\; \sum_i \; (1 + 2\beta)\, \overrightarrow{v_i}^2 - 2\beta \, \overrightarrow{v_i} \cdot \overrightarrow{v_{i+1}}. \tag{2}$$

This energy involves terms which depend on two successive values of f at the same time, and consequently the trick to remove the third loop in the complexity calculus does not work anymore, because, in the graph of f, the edges between nodes on the $j-1$-th column and the j-th column have now different costs, depending on $j-1$ and on j.

The complexity is now $m \times n^2$.

1.2.2 Final point not known

In that case, the constraint $f(m) = n$ is removed. It does not change the algorithm: at the final step, instead of computing only the best path which ends in (A_m, B_n), compute all best paths (A_m, B_j) and select the best one amongst them. Same complexity (in both case, with or without spatial coherency enforced).

1.2.3 Initial point not known

Constraint $f(1) = 1$ is removed. Unhappily this implies running the previous algorithm n times, one for each possible initialization. New complexity:

- basic dynamic time warping : $m \times n^2$
- with spatial coherency enforced : $m \times n^3$.

Remark for the first case : there exists a faster algorithm, in $n^2 \log(n)$ (confusing n with m). See [1].

1.2.4 Case of two closed curves

Pick up one point on A, called initial point A_1. Select a direction, follow the polygon along this direction and number vertices accordingly, with, at the end of the loop, $A_{m+1} = A_1$. Just apply the previously studied case "initial point not known" (section 1.2.3) on $(A_1, A_2, \ldots, A_m, A_{m+1})$, with the n possible initializations for $f(1)$ and with the final point constraint $f(m+1) = f(1)$. Same complexity : $m \times n^2$ or $m \times n^3$ depending on the use of coherence reinforcement.

1.2.5 Increasing mapping precision by oversampling

One major problem of this framework is that a vertex A_i has to be matched with a *vertex* B_j even if it would have better fitted into the middle of a segment $[B_j, B_{j+1}]$. This framework can only map points A_i onto a discrete set of predefined vertices B_j, it solves a discrete problem between two sets of polygon vertices, it does not solve a continuous problem on the whole continuous polygons. It is however easy to obtain a sub-pixel quality (supposing edges of B have pixel length).

A second major reason to do what follows is that energies like (1) involve a penalty term on the difference between the vector joining two successive points $\overrightarrow{A_i A_{i+1}}$ and the one joining their corresponding points $\overrightarrow{B_{f(i)} B_{f(i+1)}}$. The optimal solution of such an energy will depend strongly on the precise discretization of B. For example if there are more points on A than on B, for some value of i we will have $f(i) = f(i+1)$ and consequently one vector $\overrightarrow{A_i A_{i+1}}$ will be compared to $\overrightarrow{0}$. More generally, to afford such a penalty term, we need a finer discretization on B than on A.

Refine B by replacing each edge $[B_j, B_{j+1}]$ by a sequel of edges

$$[B_j, B_j^1], [B_j^1, B_j^2], [B_j^2, B_j^3], \ldots, [B_j^{p-1}, B_j^p], [B_j^p, B_{j+1}]$$

where $(B_j, B_j^1, B_j^2, \ldots, B_j^p, B_{j+1})$ is a set of $p + 2$ points regularly spaced on the segment $[B_j, B_{j+1}]$. And run the previous algorithm to map A to this oversampled version of B.

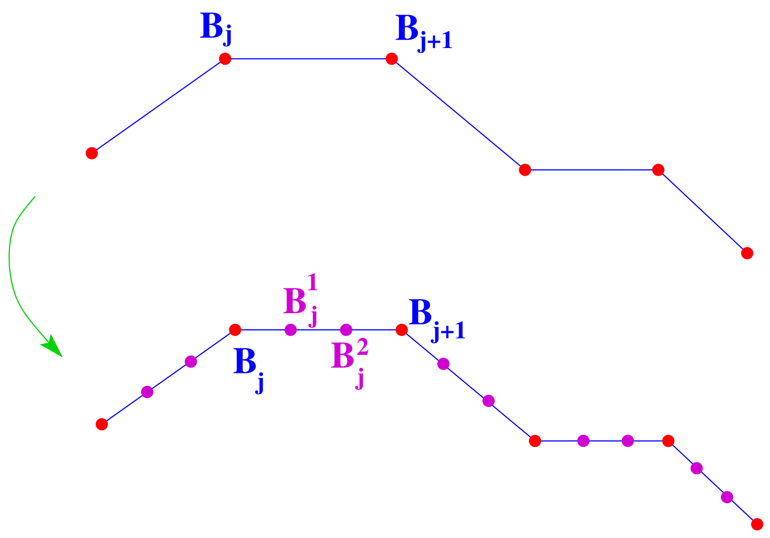


Figure 11: Oversampling so as to refine the quality of the matching.

1.3 Convergence of the solution with increasing oversampling

Let us denote by B^p the polygon B oversampled with a factor $p+1$ (i.e. with p intermediate points between all pair of consecutive vertices). Let us also denote by f_k the global minimizer of $E_k(f_k) = E(A, B^k, f_k)$. The question is : does the series of solutions f_k converge somehow when $k \to +\infty$?

Let us denote by E_∞ the energy when considering the whole polygon B, i.e. when vertices of A can be matched to any point on B. $E_\infty(f)$ is a function defined on the finite discrete set $[|1, m|]$, with continuous values in $\mathbb{S}_1$ (consider an arc-length parameterization of the closed curve B). Thus, minimizing E is a search for optimal parameters in $\mathbb{S}_1^m$.

This section can be skipped if you do not like mathematics or convergence studies, but you should pay attention at least to the last frame-box before 1.4.

1.3.1 Derivatives of E_∞

E_∞ has a quadratic form, even if not directly of f, but of $\overrightarrow{v_i} = \overrightarrow{A_i B_{f(i)}}$ (see equation 2). Hence its writes $E_\infty(f) = Q(\{\overrightarrow{v_i}\}) = {}^t\overrightarrow{v} Q \overrightarrow{v}$ where $\overrightarrow{v} = \{\overrightarrow{v_i}\}$. The quadratic form Q can be written as a $m \times m$ matrix with $1+2\beta$ on the diagonal and 2β on the first sub- and sup-diagonals:

$$Q = \begin{pmatrix} 1+2\beta & -\beta & 0 & \dots & 0 & -\beta \\ -\beta & 1+2\beta & -\beta & 0 & \dots & 0 \\ 0 & -\beta & 1+2\beta & -\beta & 0 & \dots \\ \dots & & & \ddots & & \\ 0 & \dots & 0 & -\beta & 1+2\beta & -\beta \\ -\beta & 0 & \dots & 0 & -\beta & 1+2\beta \end{pmatrix}$$

Q should be read either as $Q \otimes Id_2$, i.e. a $m \times m$ matrix of 2×2 matrices, whose elements $Q_{i,j}$, which should be read as $Q_{i,j}\, Id_2$, are proportional to the identity matrix Id_2 (acting on $\mathbb{R}^2$ vectors $\overrightarrow{v_i}$), or as the tensor product $Q \otimes Q$ acting on $\mathbb{R}^{2m} = \mathbb{R}^m \times \mathbb{R}^m = (\mathbb{R}^2)^m$ by $Q\overrightarrow{v} = Q(\overrightarrow{v_i})_x \otimes Q(\overrightarrow{v_i})_y$.
From its expression in equation (2), the matrix Q is symmetric positive definite.

The first order derivative of E_∞ is then:

$$\frac{dE_\infty}{df}(f) \;=\; \sum_i \frac{dQ}{d\overrightarrow{v_i}}(\{\overrightarrow{v_i}\}) \times \frac{d\overrightarrow{v_i}}{df}$$

Q is smooth (2-degree polynomial). $\overrightarrow{v_i}$, as a function of $f(i)$, is differentiable everywhere except on arc-lengths of geometrical vertices of the polygonal curve B (a finite number of points). However $\overrightarrow{v_i}$ is Lipschitzian (no big irregularity at these points, just the direction of moving changes). Almost everywhere, we have:

$$\frac{d^2E_\infty}{df(i)\,df(j)} \;=\; \frac{d^2Q}{d\overrightarrow{v_i}\,d\overrightarrow{v_j}} \times \frac{d\overrightarrow{v_i}}{df(i)} \times \frac{d\overrightarrow{v_j}}{df(j)} + \frac{dQ}{d\overrightarrow{v_i}} \times \frac{d^2\overrightarrow{v_i}}{df(i)^2}$$

with d^2Q being constant (i.e. does not depend on any $\overrightarrow{v_i}$ or $f(i)$), since Q is quadratic. The vector $\overrightarrow{t_B}(f(i)) = \frac{d\overrightarrow{v_i}}{df(i)}$ is the tangent vector to B at point $B_{f(i)}$, with constant speed (in norm) since we have chosen an arc-length parameterization of B. On geometrical vertices of the polygonal curve B, the quantity $\frac{d\overrightarrow{v_i}}{df(i)}$ is ill-defined but is finite. Everywhere where the tangent of the polygonal curve B is defined, $\frac{d^2\overrightarrow{v_i}}{df(i)^2} = \frac{d\overrightarrow{t_i}}{df(i)} = \overrightarrow{0}$. On the vertices, these quantities are Dirac peaks weighted with direction variations. To sum up, almost everywhere,

$$\frac{d^2E_\infty}{df(i)\,df(j)} \;=\; 2\,Q_{i,j}\; {}^t\overrightarrow{t_B}(f(i)) \cdot \overrightarrow{t_B}(f(j)) \tag{3}$$

and on a geometrical vertex B_k (i.e. subject to $f(i)$ being the arc-length l_k corresponding to vertex B_k):

$$\frac{d^2E_\infty}{df(i)\,df(j)} \;=\; 2\,\eta_{i=j}\,\delta_{l_k}(f(i))\;\; ({}^t\overrightarrow{v} \times Q)_i \cdot \left(\overrightarrow{t_B}(l_k^+) - \overrightarrow{t_B}(l_k^-)\right)$$

with $\eta_{i=j}$ equals 1 if $i = j$ and 0 otherwise, and with δ being the Dirac peak.

1.3.2 Lipschitzianity of E_∞ and energy convergence

As a quadratic function on a bounded domain of a Lipschitzian function (the arc-length parameterization of B), E_∞ is Lipschitzian (for a given polygon B), i.e. there exists a constant M so that:

$$\forall f^a, f^b \in \mathbb{R}^m, \qquad \left|E_\infty(f^a) - E_\infty(f^b)\right| \;\leqslant\; M\|f^a - f^b\|_{\mathbb{R}^m}$$

Note that an upper bound of the norm of the derivative of E (from formula (2) or (6) later) leads to $M = 2(1+4\beta)L\sqrt{m}$ where L is the maximum distance between one point of A and one point of B.

Consequence: Consider any oversampling discretization B^k, with maximum step size ε between two successive vertices. For any vector f discretized accordingly (i.e. with values on vertices of B_k), $E_k(f) = E_\infty(f)$. Furthermore, there exists a vector g_k close to the global minimizer f_∞ of E_∞ but with values on vertices of B_k, with $|g_k(i) - f_\infty(i)| \leqslant \varepsilon/2 \quad \forall i$, and consequently:

$$|E_k(g_k) - E_\infty(f_\infty)| \;\leqslant\; M\|g_k - f_\infty\| \;\leqslant\; \frac{M}{2}\sqrt{m}\,\varepsilon$$

and thus:

$$E_k(g_k) \;\leqslant\; E_\infty(f_\infty) + \frac{M}{2}\sqrt{m}\,\varepsilon.$$

Since f_k is the global minimizer of E_k,

$$E_\infty(f_\infty) \;\leqslant\; E_\infty(f_k) = E_k(f_k) \;\leqslant\; E_k(g_k) \;\leqslant\; E_\infty(f_\infty) + \frac{M}{2}\sqrt{m}\,\varepsilon.$$

Consequently,

Proposition: The global minimum of E_k (which is found by the algorithm) converges towards the global minimum of E_∞ when the discretization step size ε_k tends to 0 :

$$\min_f E_k(f) \to \min_f E_\infty(f) \qquad \text{when} \qquad k \to +\infty$$

and the error is bounded linearly by the step size:

$$\forall k, \qquad |E_k(f_k) - E_\infty(f_\infty)| \;\leqslant\; \frac{M}{2}\sqrt{m}\,\varepsilon_k \;\leqslant\; (1+4\beta)\,L\,m\,\varepsilon_k.$$

1.3.3 Finite number of local minima

Proposition: The number of local minima of E_∞ is finite, at most $(2n)^m$.

Proof : E_∞ is piece-wise quadratic and continuous.
Let us consider the continuous polygonal curve B, which, as any polygon, has a finite number n of geometrical sides $S_i = [B_i, B_{i+1}]$, and let us denote by L_i the line (B_i, B_{i+1}). Let $f = (f_1, f_2, \ldots, f_m)$ be a local minimum of E_∞. The energy E_∞ is differentiable almost everywhere (i.e., not where one of the B_{f_i} is a geometrical vertex). Since the set of possible values of each f_i is a closed, bounded set, and that a local minimum is reached at f, then, for each f_i, either B_{f_i} is a geometrical vertex, either the derivative with respect to f_i is 0. Let us first consider the case where B_{f_i} is not a geometrical vertex:

$$\frac{d}{df_i}E_\infty(f) \;=\; 2\,({}^t\overrightarrow{v} \times Q)_i \cdot \overrightarrow{t_B}(f_i) \;=\; 0$$

which reads:

$$\Big((1+2\beta)\overrightarrow{v_i} - \beta\,\overrightarrow{v_{i+1}} - \beta\,\overrightarrow{v_{i-1}}\Big)\cdot\overrightarrow{t_B}(f_i) \;=\; 0 \tag{4}$$

Since B_{f_i} is not a geometrical vertex, the tangent $\overrightarrow{t_B}(f_i)$ is constant in the neighborhood of f_i. Furthermore, since B is parameterized by arc-length, $\overrightarrow{v_i} = \overrightarrow{A_iB_{f_i}}$ varies linearly as a function of f_i in the neighborhood : if $B_{prec(f_i)}$ is the first geometrical vertex that precedes B_{f_i} (i.e. one of the extremities of the side on which B_{f_i} is), then we can write $\overrightarrow{v_i} = \overrightarrow{A_iB_{f_i}} = \overrightarrow{A_iB_{prec(f_i)}} + \overrightarrow{B_{prec(f_i)}B_{f_i}} =: \overrightarrow{w_i} + f_i\,\overrightarrow{t_B}(f_i)$, with $\overrightarrow{w_i}$ and $\overrightarrow{t_B}(f_i)$ both constant in the neighborhood of f_i. Note that the size of the "neighborhood" is quite big, it is the set of all indexes of points of the current geometrical segment.

Similarly, one can replace all $\overrightarrow{v_j}$ by some affine application $\overrightarrow{w_j} + f_j\,\overrightarrow{t_B}(f_j)$ on the segment neighborhood, provided B_{f_j} is not a vertex. If B_{f_j} is a vertex, then the "neighborhood" is restricted to the point itself. In all cases, equation (4) can be rewritten as a linear quantity of f with constant coefficients (within the product of the three neighborhoods of f_{i-1}, f_i, f_{i+1}):

$$\Big[1+2\beta\Big]\,f_i \;-\; \Big[\beta\,\overrightarrow{t_B(f_i)}\cdot\overrightarrow{t_B(f_{i+1})}\Big]\,f_{i+1} \;-\; \Big[\beta\,\overrightarrow{t_B(f_i)}\cdot\overrightarrow{t_B(f_{i-1})}\Big]\,f_{i-1} \;=\; a_i \tag{5}$$

where $a_i = -\Big((1+2\beta)\overrightarrow{w_i} - \beta\overrightarrow{w_{i+1}} - \beta\overrightarrow{w_{i-1}}\Big)\cdot\overrightarrow{t_B(f_i)}$ is constant (within the product of neighborhoods). Then, back to the original problem: if f is a local minimum, then:

$$\forall i \in [|1,m|], \qquad\qquad B_{f_i} \text{ is a vertex} \qquad \text{or} \qquad f_i \text{ satisfies (5).}$$

Let us consider the following matrix:

$$Q_t = \begin{pmatrix} 1+2\beta & -\beta\overrightarrow{t_1}\cdot\overrightarrow{t_2} & 0 & \dots & 0 & -\beta\overrightarrow{t_1}\cdot\overrightarrow{t_m} \\ -\beta\overrightarrow{t_2}\cdot\overrightarrow{t_1} & 1+2\beta & -\beta\overrightarrow{t_2}\cdot\overrightarrow{t_3} & 0 & \dots & 0 \\ 0 & -\beta\overrightarrow{t_3}\cdot\overrightarrow{t_2} & 1+2\beta & -\beta\overrightarrow{t_3}\cdot\overrightarrow{t_4} & 0 & \dots \\ \dots & & & \ddots & & \\ 0 & \dots & 0 & -\beta\overrightarrow{t_{m-1}}\cdot\overrightarrow{t_{m-3}} & 1+2\beta & -\beta\overrightarrow{t_{m-1}}\cdot\overrightarrow{t_m} \\ -\beta\overrightarrow{t_m}\cdot\overrightarrow{t_1} & 0 & \dots & 0 & -\beta\overrightarrow{t_m}\cdot\overrightarrow{t_{m-1}} & 1+2\beta \end{pmatrix}$$

where $\overrightarrow{t_i} := \overrightarrow{t_B(f_i)}$. Note that Q_t is symmetric definite positive:

$$\begin{aligned} {}^t\overrightarrow{x}\cdot Q_t\cdot\overrightarrow{x} &= (1+2\beta)\sum_i x_i^2 \;-\; 2\beta\sum_i \overrightarrow{t_i}\cdot\overrightarrow{t_{i+1}}\,x_i\,x_{i+1} \\ &= \sum_i x_i^2 \;+\; \beta\sum_i\Big(x_i^2 - 2\,\overrightarrow{t_i}\cdot\overrightarrow{t_{i+1}}\,x_i\,x_{i+1} + x_{i+1}^2\Big) \\ &= \sum_i x_i^2 \;+\; \beta\sum_i\Big(x_i\overrightarrow{t_i} - x_{i+1}\overrightarrow{t_{i+1}}\Big)^2 \\ &> 0 \quad \text{if } \overrightarrow{x} \neq \overrightarrow{0} \end{aligned}$$

This calculus also leads to the bounds:

$$\overrightarrow{x}^2 \;\leqslant\; {}^t\overrightarrow{x}\cdot Q_t\cdot\overrightarrow{x} \;\leqslant\; (1+4\beta)\,\overrightarrow{x}^2 \tag{6}$$

which implies that the eigenvalues of Q_t are between 1 and $1 + 4\beta$. From (3) it appears that the Hessian of E_∞ is twice Q_t, hence the bounds on its eigenvalues λ. A similar calculus can be performed to bound the first order derivative.

The condition "B_{f_i} is a vertex" reads "f_i is a boundary condition amongst n vertex possibilities", while the condition "f_i satisfies (5)" reads "$(Q_t \cdot \overrightarrow{f})_i = a_i$". Removing the lines and columns in Q_t which correspond to vertices let it still symmetric definite positive (it just changes the index set under the sum signs in the previous calculus) and thus, for each choice of vertex assignments to the f_i which correspond to vertices, we obtain a linear system with one unique solution (thanks to definite positivity).

For each solution, the searches have been performed along lines $(B_{prec(f_i)}, B_{next(f_i)})$ instead of along segments, so that we have *at most* one solution for each choice of assignments of f_i to geometrical vertices or segments of B. This means at most $(2n)^m$ local minima. Hence the number of local minima is finite.

Remark: With an exhaustive search on a finite set, one can compute the *exact* location of all local minima. With $(2n)^m$ cases to explore and as many linear systems to solve, it might take some time.

1.3.4 Convergence towards the global minimum

Proposition : If E_∞ admits only one global minimum, then the solution f_k also converges towards f_∞. In case of multiple global minima, property (11) still holds for one of the possible f_∞ (which may change with k).

Proof: If none of the points B_{f_∞} is a geometrical vertex of B, then E_∞ is infinitely differentiable at f_∞ and the first order derivative is 0. The second order derivative (3) can easily be shown to be symmetric definite positive, and constant in the neighborhood (whose size is named η). Considering λ_-^2 its smallest eigenvalue ($2 \leqslant \lambda_-^2 \leqslant 2 + 8\beta$ from equations (3) and (6)),

$$\exists\, \eta > 0,\ \forall\, f \text{ s.t. } \|f - f_\infty\| < \eta, \qquad E_\infty(f) \ \geqslant\ E_\infty(f_\infty) + \frac{\lambda_-^2}{2}\|f - f_\infty\|^2 \quad (7)$$

Amongst the finite set of (not global but) local minima, let f_1 be the one with the lowest value $E_\infty(f_1)$. Then $E_\infty(f_1) > E_\infty(f_\infty)$. Consider k big enough so that ε is small enough so that $Mm\varepsilon < E_\infty(f_1) - E_\infty(f_\infty)$, so that $E_k(f_k) < E_\infty(f_1)$. Start from f_k and perform a gradient descent. The energy is piece-wise quadratic, and continuous, so the path has finite length, reaching a local minimum, which can only be f_∞ (the only one lower). Following the same path but starting from f_∞, the value of E_∞ keeps increasing, and to quantify the increase we can apply the property (7) up to a length η along the path. The function $f \mapsto \|f - f_\infty\|$ is continuous along the path and its value is 0 at the beginning of the path. If $\|f_k - f_\infty\| > \eta$ then necessarily there exists one point f_p on the path where $\|f_p - f_\infty\| = \eta$, and we have

$E_\infty(f_k) \geqslant E_\infty(f_p) \geqslant E_\infty(f_\infty) + \frac{\lambda_-^2}{2}\eta^2$. Thus

$$\|f_k - f_\infty\| < \eta \qquad \text{or} \qquad E_\infty(f_k) \;\geqslant\; E_\infty(f_\infty) \;+\; \frac{\lambda_-^2}{2}\eta^2.$$

For k big enough, the second possibility does not hold anymore, and consequently we can apply formula (7) to f_k itself, to obtain:

$$\|f_k - f_\infty\| \;\leqslant\; \frac{1}{\lambda_-}\sqrt{2\big(E_\infty(f_k) - E_\infty(f_\infty)\big)} \tag{8}$$

and thus:

$$\|f_k - f_\infty\| \;\leqslant\; \frac{\sqrt{2M}}{\lambda_-}\, m^{1/4}\, \varepsilon^{1/2} \;\leqslant\; (1+4\beta)^{1/2}\, L^{1/2}\, m^{1/2}\, \varepsilon^{1/2} \tag{9}$$

for ε small enough, with ε being the discretization step size on B leading to the solution f_k. But for ε small enough, we also have

$$\|g_k - f_\infty\| \;\leqslant\; \sqrt{m}\,\frac{\varepsilon}{2}$$

hence

$$|E_k(g_k) - E_\infty(f_\infty)| \;\leqslant\; \frac{\lambda_+^2}{8}\; m\,\varepsilon^2 \tag{10}$$

where $\lambda_+^2 \leqslant 2 + 8\beta$ is the greatest eigenvalue of the Hessian of E_∞ at f_∞. Together with (8),

$$\|f_k - f_\infty\| \;\leqslant\; \frac{1}{2}\,\frac{\lambda_+}{\lambda_-}\, m^{1/2}\,\varepsilon \;\leqslant\; \frac{1}{2}\,(1+4\beta)^{1/2}\, m^{1/2}\,\varepsilon \tag{11}$$

which is a better bound than (9), for ε small enough. Both (11) and (9) still hold for *any* ε at the cost of replacing f_∞ by a local minimum, i.e., for any k, there exists a local minimum of E_∞ such that its distance to f_k is smaller than the quantities mentioned. If we index by j and denote by $f_{\infty\, loc\, j}$ the local minima of E_∞, then:

$$\forall k,\;\; \exists j, \qquad \|f_k - f_{\infty\, loc\, j}\| \;\leqslant\; \frac{1}{2}\,(1+4\beta)^{1/2}\, m^{1/2}\,\varepsilon$$

The case not studied yet is the one where one or several points of B_∞ are a geometrical vertices of B. The neighborhood of f_∞ is then piece-wise quadratic, and the number of pieces is finite. The first order directional derivatives can be computed. Contrarily to the previous case, they may be non zero. However they are necessarily non-negative (since f_∞ is a local minimum). We can re-use the previous framework with λ_-^2 the smallest eigenvalue amongst the smallest eigenvalues of the Hessians of all quadratic functions. Indeed, the property (7) still holds and the proof ends. The convergence rate is however better in this case. For each variable f_i, let us denote by d_i the lowest of the two partial directional derivatives with respect to f_i (which may differ if B_{f_i} is a vertex). When B_{f_i} is not a vertex, $d_i = 0$, but if it is, $d_i \geqslant 0$ and most often in practice d_i will be non zero. We obtain, in the neighborhood of f_∞:

$$E_\infty(f) \;\geqslant\; E_\infty(f_\infty) + \sum_i d_i |f_{k,i} - f_{\infty,i}| \;+\; \frac{\lambda_-^2}{2}\|f - f_\infty\|^2$$

and thus, with (10):

$$\sum_i d_i |f_{k,i} - f_{\infty,i}| \;\leqslant\; \sum_i d_i |f_{k,i} - f_{\infty,i}| + \frac{\lambda_-^2}{2} \|f - f_\infty\|^2 \;\leqslant\; \frac{\lambda_+^2}{8}\; m\,\varepsilon^2$$

$$\forall i \text{ s.t. } d_i \neq 0, \qquad |f_{k,i} - f_{\infty,i}| \;\leqslant\; \frac{1}{d_i}\frac{\lambda_+^2}{8}\; m\,\varepsilon^2 \;\leqslant\; \frac{1+4\beta}{8\,d_i}\, m\,\varepsilon^2.$$

If d_i is too small or 0, then the bound (11) can still be used instead.

To sum up, we have :

- Energy convergence speed :

$$\forall k, \qquad E_\infty(f_k) \;\leqslant\; E_\infty(f_\infty) \;+\; (1+4\beta)\; L\; m\; \varepsilon_k$$

- Closeness to local minima:

$$\forall k, \quad \exists\, j(A, m, B, \varepsilon_k), \qquad \|f_k - f_{\infty\, loc\, j}\| \;\leqslant\; \frac{1}{2}\,(1+4\beta)^{1/2}\, m^{1/2}\, \varepsilon_k$$

- Convergence towards the global minimum:

$$\exists\, k_0(A, m, B), \;\; \forall k > k_0, \qquad \|f_k - f_\infty\| \;\leqslant\; \frac{1}{2}(1+4\beta)^{1/2}\, m^{1/2}\, \varepsilon_k$$

- Faster convergence at vertices (i.e. $\forall i$ s.t. $B_{f_{\infty,i}}$ is a geometrical vertex of B):

$$\exists\, k_0(A, m, B), \;\; \forall k > k_0, \qquad |f_{k,i} - f_{\infty,i}| \;\leqslant\; \frac{1+4\beta}{8\,d_i}\, m\,\varepsilon_k^2$$

Note that if we divide the energies by m and the norms by $\sqrt{m}$ in order that the quantities expressed converge towards integrals when $m \to +\infty$, then the number of points m on A disappear from all inequalities, which means that the convergence speeds do not depend on the discretization of A. The convergence speed does not depend on the number n of geometrical vertices of B. However the number of local minima with values $E_\infty(f_{\infty\, loc\, j})$ close to the global one may increase with m and consequently so may $k_0(A, m, B)$. In all cases a good local minima will be guaranteed ***uniformly*** as a function of the discretization step size ε only (and of the polygon size L), neither of A, nor B: the error on the energy and the distance to the closest minimum are linearly bounded by ε.
Remark: the interpretations of the latter paragraph will have to change when later in this report we will show that we should replace β by $\frac{1}{\varepsilon_A^2}\beta$ in all expressions.

1.3.5 Continuous precision

Continuous output

It is possible to obtain a continuous precision (of the output) without considering an infinite sampling of B. Since the energy is piece-wise quadratic, the exact solution can be found provided the good piece of quadric is selected. The selection could be made thanks to the dynamic time warping algorithm with a small step size. Then either a Newton gradient descent could be performed or the linear system involving Q_t could be solved.

Continuous input

In the general case, knowing where to match two consecutive points A_i and A_{i+1} to B_{f_i} and $B_{f_{i+1}}$ respectively does not tell that the whole segment $[A_i, A_{i+1}]$ should be matched to $[B_{f_i}, B_{f_{i+1}}]$. Indeed, in some special cases, the latter could be not included in B at all... and the arc-length coordinates of B_{f_i} and $B_{f_{i+1}}$ could differ a lot.

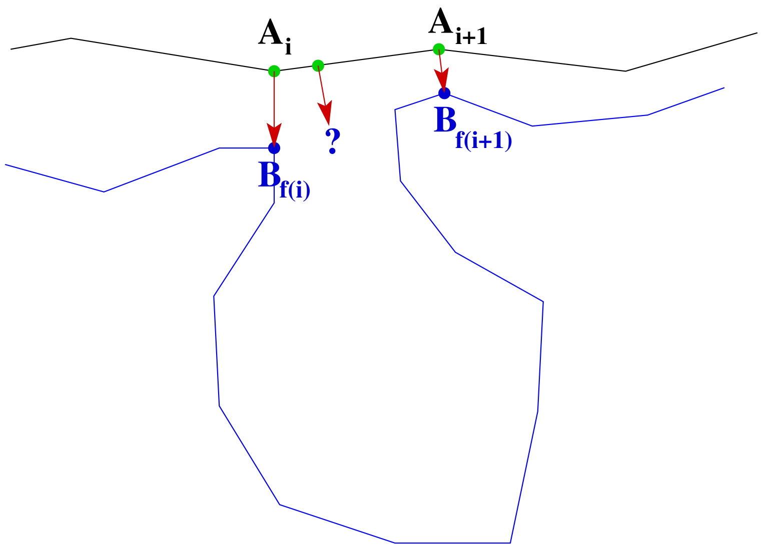


Figure 12: In the general case, no matching extension is possible. Just refine the discretization step on A or oversample it.

If B could be a relatively smooth curve, then one solution would be to consider a step size on A smaller than the distance between B and its skeleton, so that if B_{f_i} and $B_{f_{i+1}}$ are close enough, then $f(i)$ and $f(i+1)$ also and the matching extension can be performed. However, this would work only if the computed B_{f_i} and $B_{f_{i+1}}$ are close enough, which will not be the case if A and B do not match completely. And B is a polygon, so the distance to its skeleton is 0. However B may be a discretization of a smooth curve, in which case the skeleton of the smooth curve could be considered instead of B's one.

Another point of view is to consider the matching from B to A, and to reverse it. If both matchings correspond approximately, then the continuous extension makes sense. Else there is an intrinsic singularity in the matching problem between A and B which cannot be solve.

Quality of refinement

Let A and B be two smooth continuous curves with strictly positive distances to their skeletons, namely d_A and d_B respectively. By smooth we mean at least twice geometrically differentiable (i.e. with respect to their arc-length), with bounded and continuous second order derivative. Let f be a function matching A to B, i.e. which associates to the arc-length of points on A the arc-length of points on B. Then consider the following matching energy:

$$M(f) = \frac{1}{|A|} \int_A \left\| \overrightarrow{A(s)B(f(s))} \right\|^2 + \beta \left\| \overrightarrow{t_A}(s) - \left(\frac{d}{ds} f(s) \right) \overrightarrow{t_B}(f(s)) \right\|^2 ds$$

where $|A|$ is the length of A. Here f has to be continuous (and differentiable), otherwise squares of Dirac peaks appear. Note that $\left\| \overrightarrow{t_A}(s) - \left(\frac{d}{ds} f(s) \right) \overrightarrow{t_B}(f(s)) \right\| = \left\| \frac{d}{ds} \overrightarrow{A(s)B(f(s))} \right\|$. This energy has been designed so that, given two regular (uniform) discretizations A^k and B^k of A and B with maximal step sizes ε_A and ε_B respectively, the discretized version of M is the one studied before:

$$M(A^k, B^k, f^k) := \frac{1}{m} E_\infty(A^k, B^k, f^k)$$

where f^k maps vertices of A^k to any points on the polygon B^k (not necessarily to vertices). The following study will lead to a more accurate discretized energy in the case of non-uniform discretizations.

Proposition : For ε_A and ε_B small enough ($< d_A$ and d_B respectively), if the minimizer f^k of $M(A^k, B^k, f^k)$ satisfies

$$\forall i, \qquad |f^k(i+1) - f^k(i)| \leqslant \min\{K\varepsilon_A,\, d_B\}$$

then a continuous extension f from A to B can be computed from it, and it satisfies:

$$M(f) \quad \leqslant \quad M(A^k, B^k, f^k) + \alpha_1 \varepsilon_A \quad \leqslant \quad \inf_{\substack{\text{g} \\ \text{K-Lipschitz}}} M(g) \quad + \quad \alpha_2 \varepsilon_A$$

with $\alpha_1 = L\frac{1+K}{2}$ and $\alpha_2 = 2\alpha_1 + \frac{1}{d_A} + 2K(\frac{1}{d_A} + \frac{K^2}{d_B}) + \frac{2}{d_B}K^4$. Note that ε_B does not appear in the constants, indeed its role is understated by K by $\varepsilon_B \leqslant K\varepsilon_A$, and K can be arbitrarily chosen (greater than 1). Note that these inequalities are still valid when considering the minimizer of $E_k(A^k, B^k, f^k)$ up to the addition of the associated cost. Note also that in the proof we have to change the definition of the energies so that the inequalities stand for non-regular discretizations of the shapes.

Remark: the constraint $\forall i,\ |f^k(i+1) - f^k(i)| \leqslant \min\{K\varepsilon_A,\, d_B\}$ is easy to enforce in the algorithm. However, for a given K, there may be no solution.

Proof of the first inequality.
When following the contour A, one goes through vertices of A_k quite regularly. For a given point $A(s)$, let us denote by $prec(s)$ the index of the vertex of A^k which just precedes it on A. Let us denote by a_i the arc-length on A of the vertex A^k_i. Then $A(s)$ belongs to the small curve segment $A(a_{prec(s)})A(a_{prec(s)+1})$. Let us denote $prec(s)$

by i for readability purposes, and interpolate the function f^k between vertices i and $i+1$ by:

$$g(s) = f^k(i) + \frac{s - a_i}{a_{i+1} - a_{i)}} \left(f^k(i+1) - f^k(i) \right)$$

and estimate $M(g)$. We have:

$$\overrightarrow{A(a_i)A(s)} = (s - a_i)\, \overrightarrow{t_A}(a_i) \ + \ \eta_A(s)\, \overrightarrow{n_A}(a_i)$$

with $\eta_A(s) \leqslant \frac{1}{d_A}\varepsilon_A^2$ because of the bound on the curvature of A. Similarly,

$$\overrightarrow{B_{f^k(i)}B_{g(s)}} = \frac{s - a_i}{a_{i+1} - a_i} \left(f^k(i+1) - f^k(i) \right)\ \overrightarrow{t_B}(f^k(i)) \ + \ \eta_B(s)\, \overrightarrow{n_B}(f^k(i))$$

with $\eta_B(s) \leqslant \frac{K^2}{d_B}\varepsilon_A^2$ because of the bound on the curvature of B, provided that $|f^k(i+1) - f^k(i)| \leqslant \min\{K\varepsilon_A,\, d_B\}$, $\forall i$. The two last equations together give:

$$\overrightarrow{A_s B_{g(s)}} = \overrightarrow{A_{a_i} B_{f^k(i)}} + (s - a_i) \left[\overrightarrow{t_A}(a_i) + \frac{f^k(i+1) - f^k(i)}{a_{i+1} - a_i}\ \overrightarrow{t_B}(f^k(i)) \right] + \overrightarrow{\eta}(s)$$

with $\|\overrightarrow{\eta}(s)\| \leqslant \left(\frac{1}{d_A} + \frac{K^2}{d_B} \right) \varepsilon_A^2$. So that

$$\left| \int_{s=a_i}^{a_{i+1}} \|\overrightarrow{A_s B_{g(s)}}\|^2 ds \ - \ w_i \|\overrightarrow{A_{a_i} B_{f^k(i)}}\|^2 \right| \ \leqslant \ \frac{(1+K)^2}{3} \varepsilon_A^3 + L\frac{1+K}{2}\varepsilon_A^2 + O(\varepsilon_A^5)$$

with $w_i = |a_{i+1} - a_i| \leqslant \varepsilon_A$. Note that $\sum_i w_i = |A|$. We obtain:

$$\left| \frac{1}{|A|} \int_A \left\| \overrightarrow{A_s B_{g(s)}} \right\|^2 ds - \sum_i \frac{w_i}{|A|} \left\| \overrightarrow{A_{a_i} B_{f^k(i)}} \right\|^2 \right| \ \leqslant \ L\frac{1+K}{2}\varepsilon_A + O(\varepsilon_A^2)$$

Note that $\frac{w_i}{|A|} = \frac{1}{m}$ if the discretization step size is constant along A. The same way,

$$\left\| \overrightarrow{t_A}(s) - \overrightarrow{t_A}(a_i) \right\| \leqslant \frac{1}{d_A}\varepsilon_A^2$$

$$\left\| \overrightarrow{t_B}(g(s)) - \overrightarrow{t_B}(f^k(i)) \right\| \leqslant \frac{1}{d_B} K^2 \varepsilon_A^2$$

and

$$\frac{d}{ds} g(s) = \frac{f^k(i+1) - f^k(i)}{a_{i+1} - a_i}$$

so that

$$\left| \int_{s=a_i}^{a_{i+1}} \left\| \overrightarrow{t_A}(s) - \left(\frac{d}{ds} f(s) \right) \overrightarrow{t_B}(f(s)) \right\|^2 ds - (a_{i+1} - a_i) \left\| \overrightarrow{t_A}(a_i) - \frac{f^k(i+1) - f^k(i)}{a_{i+1} - a_i} \overrightarrow{t_B}(f^k(i)) \right\|^2 \right|$$

$$\leqslant \ w_i\, (1+K) \left(\frac{1}{d_A} + \frac{K^2}{d_B} \right) \varepsilon_A^2 \ + \ O(\varepsilon^4)$$

hence

$$\left| \frac{1}{A} \int_A \left\| \overrightarrow{t_A}(s) - \left(\frac{d}{ds} f(s) \right) \overrightarrow{t_B}(f(s)) \right\|^2 ds - \sum_i \frac{1}{|A| w_i} \left\| \overrightarrow{A_i A_{i+1}} - \overrightarrow{B_{f^k(i)} B_{f^k(i+1)}} \right\|^2 \right|$$

$$\leqslant \; (1+K) \left(\frac{1}{d_A} + \frac{K^2}{d_B} \right) \varepsilon_A^2 \; + \; O(\varepsilon^3)$$

Consequently :

$$\left| M(g) - F(A^k, B^k, f^k) \right| \; \leqslant \; L \frac{1+K}{2} \varepsilon_A + O(\varepsilon_A^2)$$

where

$$F(A^k, B^k, f^k) = \sum_i \frac{w_i}{|A|} \left\| \overrightarrow{A_{a_i} B_{f^k(i)}} \right\|^2 + \beta \frac{1}{|A| w_i} \left\| \overrightarrow{A_i A_{i+1}} - \overrightarrow{B_{f^k(i)} B_{f^k(i+1)}} \right\|^2 .$$

Note that $F = \frac{1}{m} E_\infty =: M(A^k, B^k, f^k)$ in case of regular discretization.

Proof of the second inequality.
Consider f the global minimizer of M if it exists, or otherwise any matching close to the optimum (any f s.t. $M(f) \leqslant \inf_g M(g) + \varepsilon$ with ε arbitrarily small). f is differentiable and defined on a compact set, so it is continuous and Lipschitzian. We will suppose here that f is also K-Lipschitzian (with K predefined). In fact the hypothesis of the proposition could be "*M admits a K-Lipschitzian solution, or the* inf *of M on the set of the K-Lipschitzian functions equals the* inf *of M*". Then let us build g^k by $g^k(i) = f(a_i)$, a candidate solution for the discretized problem $M(A^k, B^k, g^k)$. We have:

$$\forall s \in [a_i, a_{i+1}], \qquad \left| f(s) - g^k(i) \right| \; \leqslant \; K \left| s - a_i \right| \; \leqslant \; K \varepsilon_A$$

We can proceed as previously, except for $\frac{d}{ds} f(s) - \frac{g^k(a_{i+1}) - g^k(a_i)}{a_{i+1} - a_i}$ on which we have no bound. We will prove instead that an affine function between a_i and a_{i+1} does not increase the energy much, and re-use the previous framework. Consider the functional of 3 functions a, b, f:

$$G(a, b, f) = \int_{x_0}^{x_1} \left| a - b \frac{df}{ds} \right|^2 ds.$$

When a and b are constant, the minimum of G over f with fixed extremities $f(x_i) = y_i$ is achieved for the affine function f. Indeed:

$$G(a, b, f) = a^2 - 2ab \int_{x_0}^{x_1} \frac{df}{ds} ds + b^2 \int_{x_0}^{x_1} \frac{df}{ds}^2 ds$$

with $a^2 - 2ab \int_{x_0}^{x_1} \frac{df}{ds} ds = a^2 - 2ab(y_1 - y_0)$ is constant. Then we have

$$\int_{x_0}^{x_1} 1 ds \;\; \int_{x_0}^{x_1} \frac{df}{ds}^2 ds \geqslant \left(\int_{x_0}^{x_1} 1 \times \frac{df}{ds} ds \right)^2 = (y_1 - y_0)^2$$

with equality only if $\frac{df}{ds}$ is collinear to 1, i.e. is constant. We thus have found $f_{min}(s) = \frac{y_1 - y_0}{x_1 - x_0}$ which minimizes $G(a, b, f)$. Now let us add small variations η_a and η_b to G (with maximal values e_a and e_b respectively) and search for the new minimum. For all f we have:
$|G(a + \eta_a, b + \eta_b, f) - G(a, b, f)|$

$$\leqslant\ 2ae_a + e_a^2 + 2(ae_b + be_a) \int_{x_0}^{x_1} \left|\frac{df}{ds}\right| ds + (2be_b + e_b^2) \int_{x_0}^{x_1} \frac{df}{ds}^2 ds$$

$$\leqslant e_a(2a - e_a) + 2K(ae_b + be_a) + K^2 e_b(2b + e_b)$$

if f is K-Lipschitzian. Consequently the infimum values of both cannot be far:
$G(a, b, f_{min}) = \inf_{\substack{\text{g} \\ \text{K-Lipschitz}}} G(a, b, g)$

$$\leqslant \inf_{\substack{\text{g} \\ \text{K-Lipschitz}}} G(a + \eta_a, b + \eta_b, g) + e_a(2a + e_a) + 2K(ae_b + be_a) + K^2 e_b(2b + e_b).$$

Applied to $G(\overrightarrow{t_A}(a_i), \overrightarrow{t_B}(g^k(a_i)), f)$ and its variation $\int_{s=a_i}^{a_{i+1}} \left\|\overrightarrow{t_A}(s) - \left(\frac{d}{ds} f(s)\right) \overrightarrow{t_B}(f(s))\right\|^2 ds$ we obtain that the previously studied affine interpolation reaches almost the global minimum of the derivative term with fixed extremities. So:

$$\left|\int_{s=a_i}^{a_{i+1}} \left\|\overrightarrow{t_A}(s) - \left(\frac{d}{ds} f(s)\right) \overrightarrow{t_B}(f(s))\right\|^2 ds - (a_{i+1} - a_i) \left\|\overrightarrow{t_A}(a_i) - \frac{g^k(i+1) - g^k(i)}{a_{i+1} - a_i} \overrightarrow{t_B}(g^k(i))\right\|^2\right|$$

$$\leqslant \frac{1}{d_A} \varepsilon_A^2 + 2K(\frac{1}{d_A} + \frac{K^2}{d_B}) \varepsilon_A^2 + \frac{2}{d_B} K^4 \varepsilon_A^2 + O(\varepsilon_A^3).$$

Using the previous framework and summing the bounds, we obtain:

$$\left|M(f) - M(A^k, B^k, g^k)\right| \leqslant \left(L \frac{1+K}{2} + \frac{1}{d_A} + 2K(\frac{1}{d_A} + \frac{K^2}{d_B}) + \frac{2}{d_B} K^4\right) \varepsilon_A.$$

But $M(A^k, B^k, f^k) \leqslant M(A^k, B^k, g^k)$, so

$$M(A^k, B^k, f^k) \leqslant M(f) + O(\varepsilon_A).$$

Remark: this convergence study showed that we should use the coefficients $w_i = a_{i+1} - a_i = \|A_i A_{i+1}\|$ rather than the uniform ones, and minimize

$$F(A^k, B^k, f^k) = \sum_i \frac{w_i}{|A|} \left\|\overrightarrow{A_{a_i} B_{f^k(i)}}\right\|^2 + \beta \frac{1}{|A| w_i} \left\|\overrightarrow{A_i A_{i+1}} - \overrightarrow{B_{f^k(i)} B_{f^k(i+1)}}\right\|^2 .$$

However $F = \frac{1}{m} E_\infty =: M(A^k, B^k, f^k)$ in case of regular discretization. The previous convergence studies in 1.3.4 remain valid provided β is replaced by $\frac{1}{w_i^2}\beta$. Consequently, in order the bounds with β to remain bounds, the discretization of A should not only satisfy $w_i \leqslant \varepsilon_A$ but also $w_i \geqslant \alpha \varepsilon_A$ (for any fixed choice of $\alpha > 0$, for example 1/2). Then, in order to combine the bounds on ε_A with the first ones in $\varepsilon_k = \varepsilon_B$, which becomes :

$$\forall k, \qquad F_\infty(f_k) \;\leqslant\; F_\infty(f_\infty) + (1 + \frac{4}{\alpha^2} \frac{\beta}{\varepsilon_A^2})\, L\ \varepsilon_B$$

we have to require that ε_B goes faster towards 0 than ε_A^2. However this bound might be not the most optimal one. If we look at the proximity of f_k to local minima, we need only ε_B to decrease faster than ε_A.

1.4 Increasing speed by neighborhoods

To speed up the algorithm, the possible matching points B_j for A_i could be taken amongst only the nearest neighbors of A_i. If we fix a constant K and search for the indexes $\mathcal{N}_i = \{\mathcal{N}_i^1, \mathcal{N}_i^2, \dots, \mathcal{N}_i^K\}$ of the K nearest neighbors of A_i amongst B points, then the number n of points of B is replaced by K in all previous complexity computations. For example the complexity in the case of closed curve matching with coherency enforcement becomes $m \times K^3$. To this complexity should be added the one of searching for the nearest neighbors. With a naive search, it becomes $m \times (n + K^3)$.

1.5 Dealing with shapes, i.e. sets of close curves with random topology

1.5.1 A polygon to a set of polygons

First, let us match a polygon A onto a set of polygons $B = \cup_i B_i$. We proceed the same way as previously for two closed curves, except that we remove the constraint that the matching function f should be increasing. Indeed, there is no order on the set of points of a *set* of polygons, and anyway the constraint is useless in the way the algorithm proceeds. Thus, two consecutive points on A may be matched to two points on different connected components of B, but this is not a problem (it is even what we are searching for) since this would mean that these two connected components are very close at this location (and thus that they could be merged together in another frame). The spatial coherency criterion proves useful here.

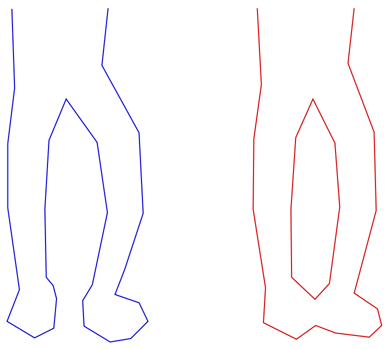


Figure 13: Topology changes appear frequently.

Also, in case of large movements of thin object parts (like fingers, or legs), we would like the left side of a leg to be matched to the left side of the leg in the other shape (and not the right one which is however closer), and a way to ensure that is to take into account the orientation of the normals (which point towards the outside of shapes) in the distance between possibly matching points.

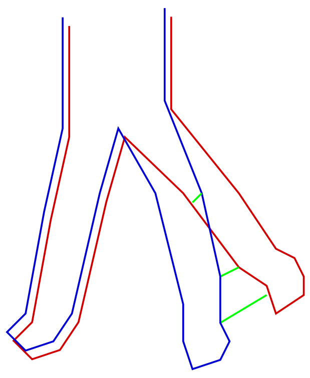


Figure 14: Nearest point on the other shape (bad match).

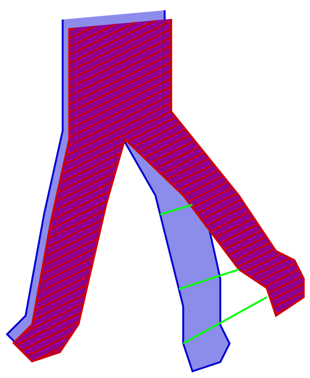


Figure 15: Nearest point, including the normal orientation (towards the outside) in the distance.

1.5.2 A polygon to a shape or void

We add another possibility of matching for any point : the matching to *void*, which means that no possible corresponding point has been found on B. Choosing *void* rather than a point of B as a corresponding point has a cost, which is higher if the previous point is not already matched to *void*. This is in order to avoid that just one point may be matched to nothing, while it may be coherent that a whole segment may not appear in the other shape. The cost should be put high enough so that matching to *void* remains exceptional (points should not be matched to *void* in case of a big deformation).

1.5.3 A shape (set of polygons) to a shape

Just match successively each of the polygons of the first shape with the previous method. Ok it might lack a bit of coherency but it allows to benefit from the locally linear structure of shapes (each vertex has two neighbors, and there is a simple way to order the set of vertices).

2 Transport of fields

2.1 Easy scheme as introduction

You are given shapes A, B and the matching function $f_{B\to A}$ from B to A. You are also given any function g_A defined on (vertices of) A and you would like to transport it to (vertices of) B. A possible solution is to consider $g_B = g_A \circ f_{B\to A}$. If $f_{B\to A}(s) = \varnothing$ then set $g_B(s) = 0$.

2.2 Vectors are not real values

The previous scheme works fine to transport real-valued functions, but when transporting vectors, one may want to rotate the vectors accordingly, so that the image of a vector which was orthogonal (or collinear) to $\overrightarrow{t_A}(s)$ is orthogonal (resp. collinear) to $\overrightarrow{t_B}\left(f_{B\to A}^{-1}(s)\right)$. Then this is the same as transporting the two real-valued functions $\overrightarrow{g_A} \cdot \overrightarrow{t_A}$ and $\overrightarrow{g_A} \cdot \overrightarrow{n_A}$. However this requires not-too-noisy boundaries (i.e., smooth tangents and normals).

2.3 Naive, diffusive barycentric transport

Now you would like to refine the transport, using a more precise matching from B to an oversampled version of A. The point of A corresponding to a vertex of B can be expressed as a barycenter of two vertices of A: $\alpha A_i + (1-\alpha)A_{i+1}$ for a certain α between 0 and 1 and a certain i which depend on s (the coordinate of the point of B). Naive transport : set

$$g_B(s) = \alpha g_A(f_{B\to A}(A_i)) + (1-\alpha) g_A(f_{B\to A}(A_{i+1})).$$

2.4 Non-diffusive barycentric transport

Now imagine you have a series of shapes A^k and you want to transport a function defined on A^0 onto A^n by transporting successively the function from A^k to A^{k+1}. Using the previous framework will end up in diffusing the function along the shapes at each step. Instead, one can compute the whole matching function $f_{A^n \to A^0} = f_{A^1 \to A^0} \circ f_{A^2 \to A^1} \circ \cdots \circ f_{A^n \to A^{n-1}}$ and then estimate directly $g_{A^0} \circ f_{A^n \to A^0}$, performing only one interpolation instead of n.

There is an easy way to define $f_{B \to A} \circ f_{C \to B}(s)$ in most cases (but which has to be changed in the other cases).

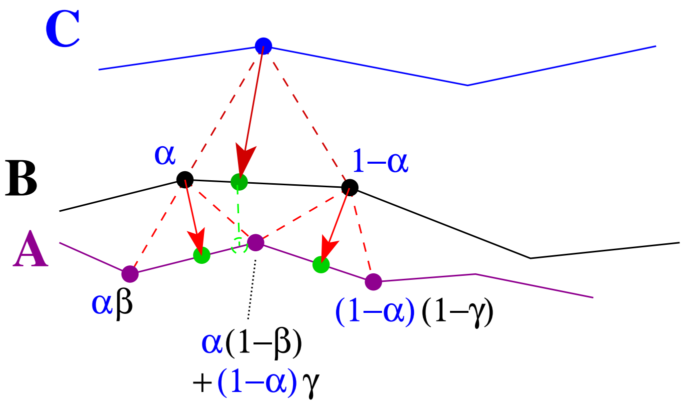


Figure 16: Simple case: compute the barycenter of the barycenters...

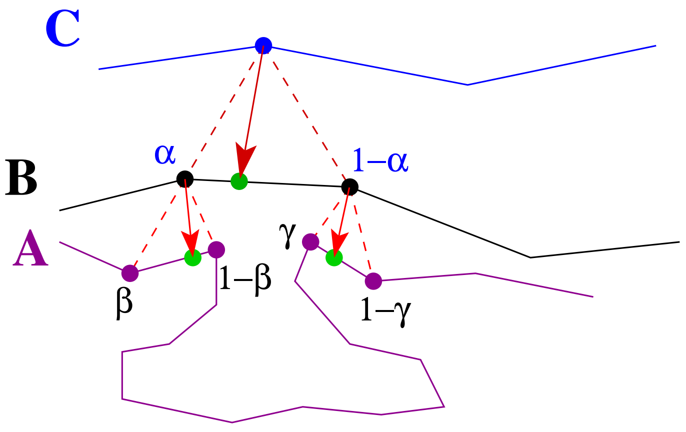


Figure 17: The other case: average the values at the two barycenters.

Let us denote by C_0 the point to be matched, by B_1 and B_2 the extremities of the segment which contains its image via $f_{C \to B}$, and A_1, A_2, A_3, A_4 the extremities of the segments which contain the images of B_1 and B_2 via $f_{B \to A}$. The points B_1 and B_2 are two consecutive points on a same connected component of B, and, the same way, A_1 and A_2 are two consecutive points of a same component of A, and so are A_3 and A_4. However A_1, A_2 and A_3, A_4 may belong to two different components of A, or be far (as a function of the arc-length). We can choose a threshold, for example ε_A, so that if the arc-length distance between A_2 and A_3 is smaller than

ε_A, then we compute the barycenter of (the arc-length of) the four points A_i with their associated weights. And then the value of g_A is estimated at this barycenter.

In the other cases, we can only compute the weighted mean of the values of g_A at the two barycenters.

We estimate the transport $T_{A_0 \to A_n}(g_{A^0})$ of the function g_{A_0} from A^0 to A^n by recursively applying the previous algorithm, avoiding as much as possible to compute averages at middle steps (only when necessary), postponing the computation of averages to the last step on A_0 as often as possible. The recursively-built set of segment extremities on $A_{n-1}, A_{n-2}, \dots, A_0$ that surround points matching a given point on A_n is a tree, with most often only two leaves (splitting steps are not frequent).

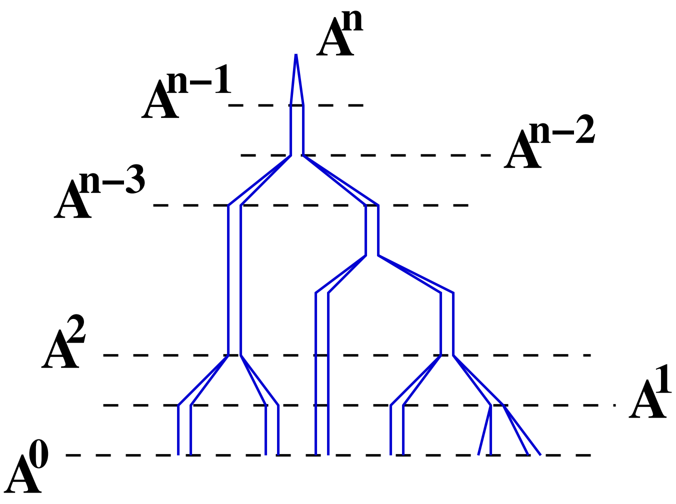


Figure 18: Following one point along the series of shapes leads to a tree, with few branches only since a split happens only in case of matching difficulty (topology change for instance).

The process has to be repeated for each point on A^n, so the complexity of the transport (knowing the matching functions $f_{A^i \to A^{i-1}}$) is $n \times m$ where m is the average number of points on shapes A_i. More simply, the computation cost is about linear in the total number of points in the set of shapes. However the cost in the *worst* case is $m \times 2^n$...

Note: we are in the barycentric framework, i.e. the matching functions $f_{A^i \to A^{i-1}}$ are computed as in part 1.2.5, from vertices of A^i to vertices of the oversampled target shape A^{i-1}.

Conclusion

Don't hesitate to ask me for more details or my code.

References

[1] F. R. Schmidt, Dirk Farin, and D. Cremers. Fast matching of planar shapes in sub-cubic runtime. In *IEEE International Conference on Computer Vision (ICCV)*, Rio de Janeiro, Brazil, October 2007.