Variance-Minimizing Transport Plans for Inter-surface Mapping

MANISH MANDAD, RWTH Aachen University and Inria, Université Côte d'Azur
DAVID COHEN-STEINER, Inria, Université Côte d'Azur
LEIF KOBBELT, RWTH Aachen University
PIERRE ALLIEZ, Inria, Université Côte d'Azur
MATHIEU DESBRUN, Caltech and Inria, Université Côte d'Azur

We introduce an efficient computational method for generating dense and low distortion maps between two arbitrary surfaces of same genus. Instead of relying on semantic correspondences or surface parameterization, we directly optimize a variance-minimizing transport plan between two input surfaces that defines an as-conformal-as-possible inter-surface map satisfying a user-prescribed bound on area distortion. The transport plan is computed via two alternating convex optimizations, and is shown to minimize a generalized Dirichlet energy of both the map and its inverse. Computational efficiency is achieved through a coarse-to-fine approach in diffusion geometry, with Sinkhorn iterations modified to enforce bounded area distortion. The resulting inter-surface mapping algorithm applies to arbitrary shapes robustly, with little to no user interaction.

CCS Concepts: • **Computing methodologies → Shape analysis**;

Additional Key Words and Phrases: Inter-surface mapping, optimal transport, Sinkhorn iterations, diffusion geometry, area distortion, conformality.

ACM Reference format:
Manish Mandad, David Cohen-Steiner, Leif Kobbelt, Pierre Alliez, and Mathieu Desbrun. 2017. Variance-Minimizing Transport Plans for Inter-surface Mapping. *ACM Trans. Graph.* 36, 4, Article 39 (July 2017), 14 pages.
DOI: http://dx.doi.org/10.1145/3072959.3073671

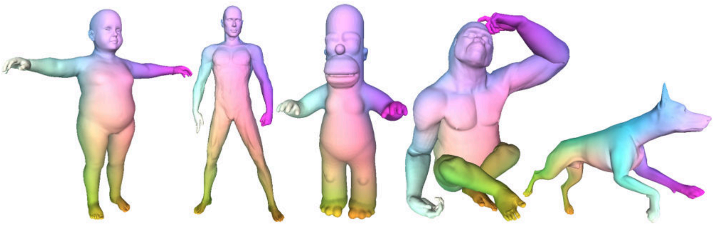


Fig. 1. **Inter-surface mapping.** Our approach generates a dense, low-distortion correspondence map between non-isometric surfaces through a geometrically-derived transport map minimizing local variances of associated neighborhoods. Only two user-defined constraints were prescribed in these models, at the tip of diametrically opposed limbs. Colors are used to indicate correspondences between all these models.

1 INTRODUCTION

Finding a map between two surfaces is a recurring task in geometry processing, required by a wide range of applications including shape matching, template fitting, attribute and animation transfer, shape completion, blending, morphing, and remeshing. It consists in establishing a meaningful correspondence map between two input surfaces, and has been referred to as inter-surface mapping [Schreiner et al. 2004], cross-parameterization [Kraevoy and Sheffer 2004], consistent parameterization [Asirvatham et al. 2005], and shape alignment in the graphics literature.

If the two surfaces are isometric, efficient algorithms exist to automatically find a locally isometric map between these shapes. When the two surfaces belong to the same specific class of shapes, maps based on semantic correspondences (where semantic features in one shape are associated to the corresponding features in the other shape) can also be constructed [van Kaick et al. 2011]; recently, approaches that learn what a user considers to be good correspondences have even been proposed [Boscaini et al. 2016]. However, in most practical cases where shapes differ significantly, there is no universal consensus on what defines the *best* correspondence map. In this more challenging case of arbitrary inputs, one typically invokes a *geometric prior* instead, such as conformality or area-preservation, to narrow down the type of suitable maps to seek.

In this paper we propose a computational approach for generating inter-surface maps that are *bijective* and have some degree of *continuity*, i.e., small neighborhoods should map to bounded neighborhoods. Our method involves the optimization of a transport map between the two surfaces, which defines an as-conformal-as-possible point-to-point map with bounded area distortion. These resulting geometric properties make our inter-surface maps particularly useful to subsequent geometry processing.

1.1 Previous Work

Previous geometric approaches can be classified by the type of mapping (point-to-point vs. soft), its properties (bijective, locally injective), its measure of distortion (isometric, conformal) and the optimization strategy employed to find the mappings. We review the most relevant approaches next.

Embedding spaces. When dealing with near isometric maps, a practical way to simplify matching is to embed the input manifolds into some target space in such a way that corresponding points of different isometric shapes are mapped to nearby points in the target space. A popular way of building such embeddings is to use spectral geometry [Bronstein et al. 2006; Gallot et al. 1994; Shtern and Kimmel 2015; Vestner et al. 2017]. Another approach in this family of methods use a common template mesh as embedding space. The base domain is often chosen to be a coarse mesh obtained [Schreiner et al. 2004] through mesh decimation , or using overlapping domains [Kraevoy and Sheffer 2004] bounded by consistent paths connecting

DOI: http://dx.doi.org/10.1145/3072959.3073671

feature points . The mapping is then constructed by optimizing a mutual tessellation of the two surfaces. While this approach can handle non-isometric transformations, decreasing the global non-linear distortion through vertex-by-vertex updates of the map is often very slow. Instead, our approach directly optimizes a transport plan over the two surfaces, bypassing the need for embedding spaces.

Parameterization. Surface mappings can be computed via mesh parameterization as well. Recent approaches offer to construct bounded distortion maps [Lipman 2012], angle-preserving maps with as uniform as possible conformal factors [Aflalo et al. 2013], using cone manifolds [Myles and Zorin 2013] or orbifolds [Aigerman and Lipman 2015], or even propose to relax the bijective condition of the parameterization to local injectivity [Aigerman et al. 2014]. One main limitation of these parameterization-based approaches is that the map distortion is not directly measured between the two input surfaces, but through a common base domain, which leads to suboptimal inter-surface maps. Mapping to a common domain also often requires designing a cut-graph which increases distortion as well (see [Aigerman et al. 2015] for a recent improvement). In contrast, our approach directly optimizes a surface-to-surface map without resorting to an intermediate domain. Moreover, while Aflalo et al. [2013] sought conformal maps with no (or bounded) area distortion, we look instead for area-preserving (or bounded area distortion) maps that are as conformal as possible.

Soft maps and optimal transport. While point-to-point mapping between surfaces has been favored in the past, soft maps [Solomon et al. 2012] and functional maps [Ovsjanikov et al. 2012] (where functions or distributions are mapped between surfaces) have been shown relevant to the analysis of correspondences through linear algebra tools. Using a mass transport formulation, Solomon et al. [2012] generate transport maps that are continuous, faithful to geometric descriptors and *soft*. Continuity of such maps means that nearby regions on one surface should map to nearby mass distributions on the other surface, where "nearby" is to be understood with respect to the geodesic earth mover's distance. In addition, softness of the map is efficiently optimized by L_2 maximization of the spreading of the probabilities in the mapping matrix describing the mass transport plan. The manner in which a soft map transports one distribution to another can also help control map continuity [Solomon et al. 2013]. Corman et al. [2015] find inter-surface mappings by integrating vector fields so as to minimize a soft map based energy that favors isometry. Note that softness of transport maps can also be used as a means to accelerate the computation of optimal transport plans: Cuturi devised an entropic regularization of transport which reduces optimal transport computational times by several orders of magnitude, at the price of smoothed-out transport plans [Cuturi 2013; Solomon et al. 2015]. Frogner et al. [2015] further extended smoothed transport to unnormalized measures, a concept also referred to as relaxed or unbalanced transport [Chizat et al. 2015, 2016]. In sharp contrast, our approach avoids soft maps altogether, and entropic regularization is used sparingly and for numerical stability only; computational scalability is, instead, gained by optimizing the transport plan in a coarse to fine fashion in diffusion space (i.e., based on a small fraction of eigenvectors of the Laplace operator).

Gromov-Wasserstein distances. Mémoli [2011] pioneered the use of the Gromov-Wasserstein distance for deriving a metric approach to the shape matching problem. Solomon et al. [2016] computed correspondences through an entropy regularized Gromov-Wasserstein objective function, using the Softassign algorithm [Rangarajan et al. 1999]. Their objective function tries to pair points with similar geodesic distances on the two shapes. However, the solver involved in this approach loops through compute-intensive iterations of cubic complexity in the number of points to be matched, and a matrix of pairwise geodesic distances must be stored, rendering the construction of a dense map intractable. In addition, the resulting maps in the Gromov-Wasserstein approach can have one-to-many correspondences (see Fig. 20). While this property may be valuable in specific applications, it does not provide the type of one-to-one mappings that geometry processing tools typically requires. We deviate from their approach by formulating a very different notion of distortion, related to the symmetric Dirichlet energy of transport maps, that naturally induces point-to-point maps. Moreover, our mapping algorithm has nearly linear complexity, thus allowing the construction of inter-surface maps over two orders of magnitude faster in practice than these existing approaches.

1.2 Contributions

In this work we introduce an approach for finding inter-surface maps based on a mass transport formulation that minimizes the variance of the images of local neighborhoods. Variance-minimizing transport plans, which we show to be minimizers of the symmetric Dirichlet energy of maps in the continuous limit, inherently favor conformality and induce continuity and sharpness of the resulting map while enforcing bounded area distortion. Although this new formulation becomes non-convex and hence substantially more challenging to minimize, we reformulate it as a biconvex minimization problem. A coarse-to-fine approach based on a hierarchy built in diffusion geometry is then devised to efficiently optimize the map through alternating minimizations and reduce the risk of getting trapped in local minima. We demonstrate our robust and scalable algorithm for inter-surface mapping on a series of non-isometric models (see, e.g., Fig. 1).

2 APPROACH

Our approach stems from the simple idea that given two pointsets $\mathbf{X}=\{x_i\}$ and $\mathbf{Y}=\{y_j\}$ sampling two input surfaces, we should aim at constructing a map between $\mathbf{X}$ and $\mathbf{Y}$ that associates neighborhoods from one pointset with neighborhoods in the other pointset, and vice-versa. We formulate this inter-surface mapping as an optimal mass transport problem between the two surfaces that minimizes the variance of the image of each neighborhood.

2.1 Setup

In order to formulate our map optimization problem, we introduce a few key concepts that we will leverage throughout our exposition.

Surfaces as distributions. Our approach considers each pointset as a *discrete mass distribution*. More specifically, we discretize the normalized area measures of the two surfaces as weighted sums of Dirac measures centered at input points, i.e., $\mu = \sum m_i \, \delta_{x_i}$ and

$\nu = \sum n_j \, \delta_{y_j}$, where the masses m_i and n_j are normalized so that each surface has unit total mass, i.e., $\sum_i m_i = \sum_j n_j = 1$. The mass of each point is assigned based on the estimated local Voronoi area of the surface it covers to account for non-uniform sampling.

Maps as transport plans. Given two distributions μ and ν, a transport plan π between them is by definition a matrix of size $|\mathbf{X}|\times|\mathbf{Y}|$ whose marginals equal μ and ν. Each entry π_{ij} represents the amount of mass transported from $x_i \in \mathbf{X}$ to $y_j \in \mathbf{Y}$. A transport plan π has two associated *transfer operators* $\pi_{\mathbf{X}}$ and $\pi_{\mathbf{Y}}$ defined as

$$\pi_{\mathbf{X}}(\delta_{x_i}) = \frac{1}{\sum_j \pi_{ij}} \sum_j \pi_{ij} \, \delta_{y_j} \; , \; \pi_{\mathbf{Y}}(\delta_{y_j}) = \frac{1}{\sum_i \pi_{ij}} \sum_i \pi_{ij} \, \delta_{x_i} .$$

Note that the denominators are respectively equal to m_i and n_j. These two linear operators describe how mass distributions on $\mathbf{X}$ are mapped to mass distributions on $\mathbf{Y}$ with the same total mass, and vice-versa. Therefore, they contain essentially the same information as the transport plan.

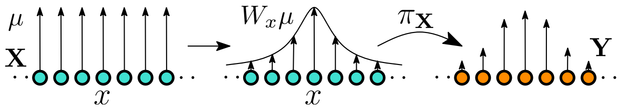
Weighting functions. For every point x on a surface, we associate an isotropic weighting function W_x centered at x (defined as either a heat kernel based function in Sec. 3.2 or a Gaussian function in Sec. 4.2) for two purposes: first, its finite support defines a notion of local neighborhood; second, weighting functions can be used as local test functions to measure the variance of an inter-surface map, as we explain next.

2.2 Variance-minimizing Transport Plans

Our approach consists in computing the transport plan π between μ and ν that minimizes the following variance-based transport cost:

$$E\,(\pi) = \int_{\mathbf{X}} \operatorname{var} \pi_{\mathbf{X}} \left(\frac{W_x \mu}{\operatorname{mass}(W_x \mu)} \right) d\mu(x) \, + \int_{\mathbf{Y}} \operatorname{var} \pi_{\mathbf{Y}} \left(\frac{W_y \nu}{\operatorname{mass}(W_y \nu)} \right) d\nu(y). \tag{1}$$

That is, this cost evaluates the integral of the variance of the image, by the transfer operator and its inverse, of the area measure modulated by local weighting functions (see inset below: (left) original measure, then (middle) modulated by weighting functions, and (right) its corresponding measure through the plan).



Properties. The reason for targeting such a variance-minimizing plan is two-fold. First, penalizing the variance of images favors transport plans corresponding to actual point-to-point maps rather than soft, diffused maps: any point mass that is mapped to a spread-out distribution incurs a significant cost. Second, variance-minimizing maps tend to be bijective, since we also account for the inverse map in the cost evaluation. Bi-continuity is also expected of the minimizers of our cost as we penalize variances of images of small neighborhoods defined by the support of our weighting functions. Moreover, we will show that our variance-based formulation of the transport cost is amenable to efficient plan optimization.

Discrete symmetric Dirichlet energy. The cost function $E(\pi)$ of a plan π can be understood a discrete approximation of the symmetric Dirichlet energy of the map and its inverse: it coincides with the Dirichlet energy for infinitesimal weighting functions with isotropic covariance matrices tending to zero. More specifically, let $f : S \mapsto S'$ be a diffeomorphism between two surfaces. Choose the weighting function at $x \in S$ to be a Gaussian distribution with covariance $\sigma^2 I$, where σ is a small positive number. Let v be a random vector drawn from this distribution. Then the integrand of E at x is by definition the variance of $f(v)$. For small σ, we may further approximate locally the two manifolds by their tangent spaces. Under this approximation $f(v)$ is a Gaussian with covariance $\sigma^2 (Df)(Df)^\top$, hence $\operatorname{var}(f(v)) = 2\sigma^2 ||Df||_F^2$. As a consequence, the total cost is proportional to the sum of the Dirichlet energy of f and the Dirichlet energy of its inverse. Note that the same conclusion holds for non Gaussian weighting functions, as long as they have isotropic covariances with total variance independent of x, and tending to 0. While generalization of the Dirichlet energy to transport plans have been proposed in the literature (see, e.g., [Solomon et al. 2013]), we are not aware of any existing formulation that also penalizes soft maps, a property that is essential for our purposes as it implies that our optimal plans can be expected to be close to conformal if no other constraints are imposed.

2.3 Optimizing Transport Plans

While the formulation of transport cost given above induces good geometric properties for the optimal map, it involves a non-convex functional which may seem difficult to minimize efficiently and robustly. We can, however, introduce auxiliary variables within the variance terms so that finding a variance-minimizing map amounts to solving a biconvex problem, which we achieve through *alternating convex minimization.*

Reformulation with auxiliary variables. Given a point x and a measure $\mu = \sum \mu_i \, \delta_{x_i}$, we define the variance of μ *with respect to* x as:

$$\operatorname{var}(\mu, x) = \textstyle\sum_i \; \mu_i \, d(x_i, x)^2, \tag{2}$$

where distances $d(\cdot,\cdot)$ are computed in a (possibly high-dimensional) Euclidean embedding of the surface $\mathbf{X}$ that μ samples, such that the minimum of this variance is found through a simple linear average

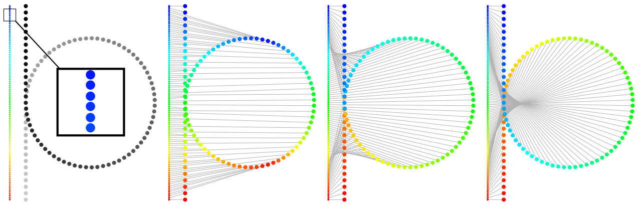


Fig. 2. **Mapping via variance minimization.** Left: Two input pointsets forming a line and a curve with a loop respectively (rainbow and grey colormaps depict their geodesic connectivity). Middle left: initial optimal transport map using the 2-Wasserstein distance (correspondences in thin lines). Middle right: our variance-minimizing map after alternating minimization iterations using ambient, Euclidean distances. Right: result after minimization using *geodesic* (i.e., intrinsic) distances instead.

of positions in this embedding. Note that using geodesic distances in the definition of the variance would be computationally intractable beyond the case of curves, while using $\mathbb{R}^3$ distances would be poorly adapted as it only provides information on the extrinsic structure of the surface; we will thus use a “diffusion” embedding (see Sec. 3.1) which is fully intrinsic, but as simple to evaluate as the $\mathbb{R}^3$ distance—and short distances in this embedding will be good approximations of geodesic distances on **X**, which enforces that our approximation will be accurate for fine enough sampling. As we will see later, these embedding can further be used in a multiscale way, which will lead to better optima and increased efficiency as well. With this notion of variance w.r.t. a point location, we can now introduce a set of point locations η_{x_i} and η_{y_j}, one for each point x_i and y_j on the two surfaces. These auxiliary variables, which we call *centers* for reasons that will become clear soon, allow us to rewrite our transport cost between **X** and **Y** as a function of both the transport plan π and the set of all centers η:

$$C(\pi,\eta) = \sum_i m_i \text{ var}\left[\pi_{\mathbf{X}}\left(\frac{W_{x_i}\mu}{\sum_k m_k W_{x_i}(x_k)}\right), \eta_{x_i}\right] + \sum_j n_j \text{ var}\left[\pi_{\mathbf{Y}}\left(\frac{W_{y_j}\nu}{\sum_k n_k W_{y_j}(y_k)}\right), \eta_{y_j}\right]. \tag{3}$$

Note that minimizing this cost functional with respect to η assigns each center η_{x_i} (resp., η_{y_j}) to the (weighted) center of mass of the image of the normalized weighting functions W_{x_i} (resp. W_{y_j}) by the transport plan π. For such values of η, the cost functional is precisely equal to the energy defined in Eq. (1).

Minimization through alternating convex problems. This augmented cost functional $C(\pi,\eta)$ is now amenable to an alternating minimization similar to Expectation-Maximization (EM): for a fixed transport plan π, minimizing $C(\pi,.)$ is efficiently achieved by relocating the centers to the barycenters of the images of the weighting functions by π (see Sec. 3.2); for fixed centers, minimizing $C(.,\eta)$ requires the computation of an optimal transport plan under the usual mass preservation constraints. Alternately minimizing the cost C over π_{ij} and η as described in Alg. 1 is a robust and efficient way to treat the original non-linear problem since both minimization problems are convex. Fig. 2 shows the transport plan after alternating minimization iterations for an example between two curves, where we use a Gaussian weighting function as our local test function to measure variance. Note that the number of iterations needed to properly “unfold” one curve onto the other depends on the width of this function: if the weighting function used as a test function is too narrow, the variance is only measured locally and many iterations are needed as Fig. 3 shows. This observation calls for a coarse to fine treatment where alternating minimizations are performed at multiple scales to improve efficiency, as we detail later.

Additional control over the mapping. The cost we defined in Eq. (3) measures the regularity of a transport plan via its local variance. However, user control over the final mapping is often desirable: the user may want to prescribe a few correspondences manually. If x_i must map to y_j, we simply constrain the center η_{x_i} to be y_j and η_{y_j} to be x_i. Additionally, one can add to the transport cost a small amount of the standard 2-Wasserstein distance between the surfaces to help the solver disambiguate symmetries, see Sec. 5.

Algorithm 1 Map optimization through alternating minimization

```
function ALTERNATING MINIMIZATION
    repeat
        $\eta \leftarrow \min C(\pi,.)$ // relocate centers to local barycenters
        $\pi \leftarrow \min C(.,\eta)$ // solve optimal transport problem
    until $\Delta C/C < 0.001$
    return $\pi$                ▹ variance-minimizing transport plan
```

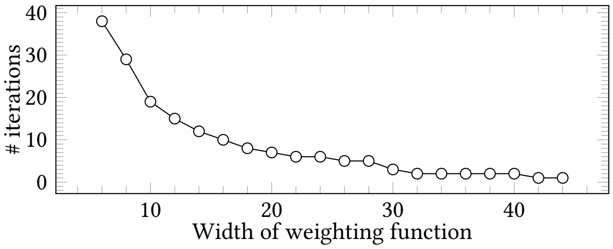


Fig. 3. **Convergence of alternating minimizations.** This plot indicates the number of iterations to reach convergence for the curve-to-curve map in Fig. 2, against the weighting function’s width (indicated as the number of neighbors x_i such that $W_x(x_i) > 0.5$) used to measure variance.

3 NUMERICAL TOOLS

Our geometric mapping algorithm relies on the frequent evaluation of distances and barycenters on surfaces, as well as the repeated evaluations of optimal transport plans. Before delving into the details of our algorithm, we discuss a few numerical tools we will rely on heavily; in particular, the use of diffusion geometry and Sinkhorn iterations will help make our approach both robust and efficient.

3.1 Diffusion Geometry

The concept of *diffusion geometry* introduced in [Coifman et al. 2005; Nadler et al. 2005] proposes to embed a Riemannian manifold into a “diffusion space” whose coordinates are derived from the eigenvectors and eigenvalues of the celebrated Laplace-Beltrami operator. In our context, this intrinsic diffusion operator Δ is approximated on each pointset and we denote its M smallest eigenvalues by $\{\lambda_k\}$ and its corresponding eigenfunctions by $\{\phi_k\}$ with $\Delta\phi_k = \lambda_k\phi_k$. Then a point $x \in \mathbf{X}$ is projected into (truncated) diffusion space as $\hat{x}$ through:

$$x \xrightarrow{\Phi_t} \hat{x} = \Phi_t(x) = \begin{pmatrix} e^{-\lambda_1 t/2}\phi_1(x) \\ e^{-\lambda_2 t/2}\phi_2(x) \\ \vdots \\ e^{-\lambda_M t/2}\phi_M(x) \end{pmatrix}, \tag{4}$$

where t controls the diffusion time scale. For two points x_1 and x_2, the corresponding heat kernel $K_t(x_1,x_2)$ is expressed as:

$$K_t(x_1,x_2) = \Phi_t(x_1)^\top \Phi_t(x_2), \tag{5}$$

while the diffusion distance between these points is computed as:

$$d_{\text{diff}}(x_1,x_2) = \|\Phi_t(x_1) - \Phi_t(x_2)\|, \tag{6}$$

or $d^2_{\text{diff}}(x_1,x_2) = K_t(x_1,x_1) + K_t(x_2,x_2) - 2K_t(x_1,x_2)$ equivalently. For small t, diffusion distances can be shown to approximate *local* *geodesic distances* very well, while increasing t has a smoothing effect by diminishing the contribution of high frequency eigenfunctions [Coifman et al. 2005]. In order to keep the computational cost low, we select only the first M smallest eigenvalues of the Laplace-Beltrami operator ($M = 100$ in practice, unless indicated otherwise).

3.2 Leveraging Diffusion Geometry

One of the key points of our approach is the use of diffusion geometry to approximate a number of important geometric notions.

Diffusion centers. The geodesic barycenter of a weighted neighborhood defined on a surface is in general not expressible in closed form and may not even be unique if the neighborhood is large. A crucial reason for using diffusion distance instead is that we can compute weighted diffusion barycenters in closed form as *simple weighted averages* of points in diffusion space as discussed in Sec. 2.3. This property brings significant speedup since converting a point to diffusion space is done efficiently via Eq. (4). Moreover, because of their intrinsic nature, diffusion coordinates are much preferable to Euclidean coordinates in $\mathbb{R}^3$ as they are more robust to averaging of distant points. These diffusion centers will thus be used in lieu of the true geodesic centers.

Weighting Functions. If distances and variances are computed in diffusion space, a consistent choice for the weighting function W_x is to rely upon the heat kernel in Eq. (5) in order to define an approximate geodesic neighborhood. While this choice is nicely isotropic for small diffusion times t, its support becomes off-centered for large values of t (see Fig. 4). We use instead the weighting function:

$$W_x(x_i) = \frac{2K_t(x,x_i)}{K_t(x,x) + K_t(x_i,x_i)}, \tag{7}$$

which remains localized and centered for any diffusion time while still satisfying $W_x(x) = 1$. In addition, we truncate the weighting function below $\varepsilon = 0.5$ (Fig. 4) in order to further localize its associated geodesic neighborhood, which will reduce computational times required to evaluate geodesic variances.

Projection to tangent planes. As part of our algorithm, local projections to tangent planes will be necessary in Section 4.4 to check map consistency. While this is easily achieved for a fine sampling of the surface in Euclidean space, the use of diffusion geometry requires the same operation to be performed directly in diffusion space. To find the two tangent vectors forming the diffusion tangent plane at a point x on the surface, we proceed as follows. We first assemble the matrix of size $M{\times}3$ for which the k-th row is $\nabla\phi_k(x)$. Then, for two approximate tangent vectors u and v in Euclidean 3D space (e.g., computed based on two nearest neighbors of x), we multiply this matrix by each of these vectors to get the diffusion tangent vectors $\hat{u}$ and $\hat{v}$, that we further orthonormalize. Any point $\hat{z}$ in diffusion space can now be directly projected onto the 2D diffusion tangent plane at x through $(\hat{z}^\top\hat{u}, \hat{z}^\top\hat{v})$. Note that this simple projection to the diffusion tangent plane through inner products provides a consistent definition of the notion of linear approximation of the surface even for coarse sampling, adding significant robustness: defining a tangent space in Euclidean space for a coarse sampling of a surface would be much less reliable.

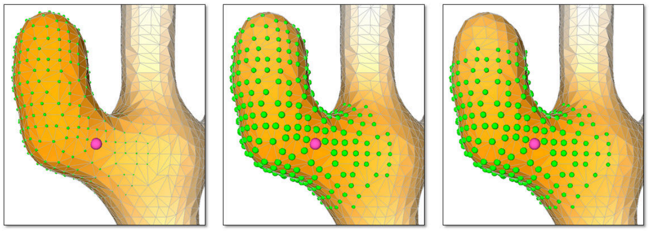


Fig. 4. **Choice of weighting functions.** From left to right: $K_t(x,x_i)$, $K_t(x,x_i)/\sqrt{K_t(x,x)K_t(x_i,x_i)}$ and $2K_t(x,x_i)/(K_t(x,x)+K_t(x_i,x_i))$ for x indicated as the purple vertex. Its neighborhood, i.e., vertices x_i for which $W_x(x_i) > \varepsilon$, is shown with green spheres proportional to function values.

3.3 Sinkhorn Iterations

Since our approach relies on repeated optimizations of transport plans, an efficient solver for optimal transport problems is mandatory. In order to avoid having recourse to Linear Programming, recent work [Benamou et al. 2015; Cuturi 2013] proposed an entropic regularization of the optimal transport problem which minimizes the Kullback-Leibler (KL) divergence:

$$\min \left\{ \mathrm{KL}(\pi|\mathcal{K}_\gamma) \overset{\text{def.}}{=} \iint \pi \left[\ln \frac{\pi}{\mathcal{K}_\gamma} - 1 \right] dxdy \right\}, \tag{8}$$

where γ controls the amount of regularization and $\mathcal{K}_\gamma$ denotes the kernel associated to the cost function. This regularized optimal transport map can be efficiently computed using Sinkhorn's matrix row/column normalization algorithm which is several orders of magnitude faster than solving a linear program.

Sinkhorn algorithm. Following [Benamou et al. 2015; Solomon et al. 2015], given a matrix $\mathbf{H}$ discretizing the kernel $\mathcal{K}_\gamma$, the transport plan minimizing Eq. (8) between two datasets $\mathbf{X}$ and $\mathbf{Y}$ is of the form $\pi = \mathbf{D_m D_v H D_w D_n}$, for diagonal matrices $\mathbf{D_v}$ and $\mathbf{D_w}$ defined by vectors $\mathbf{v}, \mathbf{w}$ that satisfy:

$$\begin{cases} \mathbf{D_v H D_w n} = \mathbf{1}_{|\mathbf{X}|}, \\ \mathbf{D_w H^\top D_v m} = \mathbf{1}_{|\mathbf{Y}|}, \end{cases} \tag{9}$$

where $\mathbf{H}$ is of size $|\mathbf{X}| \times |\mathbf{Y}|$, $\mathbf{m}$ and $\mathbf{n}$ represent the mass (normalized Voronoi area) vectors of $\mathbf{X}$ and $\mathbf{Y}$, and $\mathbf{1}_k$ is the vector of ones of size k. Notice that the properties $\pi\mathbf{1}_{|\mathbf{Y}|} = \mathbf{m}$ and $\pi^\top\mathbf{1}_{|\mathbf{X}|} = \mathbf{n}$ are enforced by Eq. (9). We find $(\mathbf{v}, \mathbf{w})$ via *Sinkhorn iterations* [Sinkhorn 1964] as detailed in Alg. 2, where $\oslash$ and $\otimes$ denote entrywise division and multiplication, respectively. Convergence is assumed when the change in KL divergence is less than 0.1%.

Algorithm 2 Sinkhorn iterations to solve for optimal transport

```
function SINKHORN
    $\mathbf{v}, \mathbf{w} \leftarrow \mathbf{1}$
    while !converged do
        $\mathbf{v} \leftarrow \mathbf{1}_{|\mathbf{X}|} \oslash \mathbf{H}\ (\mathbf{n} \otimes \mathbf{w})$
        $\mathbf{w} \leftarrow \mathbf{1}_{|\mathbf{Y}|} \oslash \mathbf{H}^\top(\mathbf{m} \otimes \mathbf{v})$
    return $\pi = \mathbf{D_m D_v H D_w D_n}$                ▷ Transport plan
```

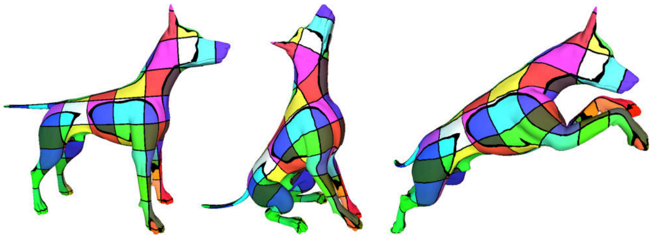


Fig. 5. **Mapping between isometric poses.** Result of our algorithm between three different dog poses; the reference pose is on the left, and its strip-coloring is displayed on the other two shapes based on resulting maps.

Working in log space. The original Sinkhorn approach assembles the kernel matrix $\mathbf{H}$ through its elements $\mathbf{H}_{ij} = \exp(-\text{coef}(\pi_{ij})/\gamma)$, where $\text{coef}(\pi_{ij})$ refers to the coefficient of π_{ij} in the cost function (3). The regularization coefficient γ used to compute the optimal transport is recommended to be set such that $\max\{\text{coef}\left(\pi_{ij}\right)\}/\gamma < 200$ [Cuturi 2013] in practice, to prevent numerical blowups due to limited floating-point dynamic range. However, this choice of coefficient usually creates a diffuse transport plan, conflicting with our objective to construct variance-minimizing plans. In our approach, we sidestep this numerical limitation by storing the coefficients of the kernel matrix in log scale and compute the Sinkhorn iterations in log scale directly (see also the concurrent work of [Schmitzer 2016] for a similar approach). This implies that we now store $\mathbf{H}_{ij} = -\text{coef}(\pi_{ij})/\gamma$, and perform the inner products $\mathbf{a}^\top\mathbf{b}$ in Steps 4-6 of Alg. 2 (involving columns or rows of $\mathbf{H}$) by first computing the maximum element $Z = \max_i\{\mathbf{a}[i] + \mathbf{b}[i]\}$, and evaluating

$$\mathbf{a}^\top\mathbf{b} := Z + \log\sum_j \exp(\mathbf{a}[j] + \mathbf{b}[j] - Z).$$

Working in log scale and subtracting the maximum element drastically reduces the negative effects of a limited floating-point dynamic range, allowing us to safely set to $\gamma = \max\{\text{coef}\left(\pi_{ij}\right)\}/10^{-4}$, a 50-fold decrease in regularization. In order to further improve efficiency, we also use the previous values of $\mathbf{v}$ and $\mathbf{w}$ instead of initializing them to $\mathbf{1}$ in each successive relaxation iterations.

3.4 Mass-Relaxed Sinkhorn Iterations

When dealing with a very non-isometric pair of shapes, enforcing strict mass preservation leads to mappings with high anisotropic differentials. This anisotropy may become significant enough locally to render the optimal map ill-suited for further geometry processing. Relaxing the mass preservation constraint helps remove this issue, while providing a simple way to bound area distortion.

Adapting Sinkhorn iterations. A way to relax the mass preservation constraint in the framework of entropy-regularized optimal transport is to solve the following optimization problem:

$$\min_{\pi\in\Pi} \text{KL}(\pi|\mathcal{K}_\gamma)$$

where Π denotes the set of matrices satisfying:

$$\begin{array}{ll} \pi_{ij} \geq 0, & \alpha_i m_i \leq \sum_j \pi_{ij} \leq \beta_i m_i, \\ \sum_{i,j} \pi_{ij} = 1, & \alpha'_j n_j \leq \sum_i \pi_{ij} \leq \beta'_j n_j, \end{array}$$

where $0 \leq \alpha_k \leq 1 \leq \beta_k$ and $0 \leq \alpha'_k \leq 1 \leq \beta'_k$ are the prescribed lower and upper bounds on the allowed mass distortion on each surface. We write the convex set of allowed solutions as the intersection of two convex sets, namely the set Π_1 of matrices satisfying the first, second and third conditions above, and the set Π_2 of matrices satisfying the first, second and fourth conditions. Optimizing over their intersection can be done using *Dykstra's algorithm* adapted to KL divergence in [Bauschke and Lewis 1998; Benamou et al. 2015]. This algorithm only requires projections of distributions on each convex set (in the sense of KL divergence), which we derive in App. A. It turns out that the KL projections on Π_1 and Π_2 boil down to row and column rescaling respectively; because of these particularly simple forms, the auxiliary variables used in the original Dykstra's algorithm do not play any role, and the overall algorithm reduces to iterated KL projections on Π_1 and Π_2. Also, since only row and column rescaling is used, we may represent the transport plan π at each iteration as a product of matrices associated to vectors $\mathbf{v}$ and $\mathbf{w}$ as in the usual Sinkhorn algorithm. Pseudocode for the algorithm is given in Alg. 3, and related functions are in Alg. 4.

Algorithm 3 Mass-Relaxed Sinkhorn iterations

```
function Mass_Relaxed_Sinkhorn
    while !converged do
        $\mathbf{lb} \leftarrow \alpha \oslash \mathbf{H}(\mathbf{n} \otimes \mathbf{w})$    ▷ lower bound
        $\mathbf{ub} \leftarrow \beta \oslash \mathbf{H}(\mathbf{n} \otimes \mathbf{w})$    ▷ upper bound
        $r \leftarrow$ solution to Global_Mass_v($\mathbf{lb}, \mathbf{ub}$)
        $\mathbf{v} \leftarrow$ Update_Vector_Global_Mass($r, \mathbf{lb}, \mathbf{ub}$)
        $\mathbf{lb} \leftarrow \alpha' \oslash \mathbf{H}^\top(\mathbf{m} \otimes \mathbf{v})$
        $\mathbf{ub} \leftarrow \beta' \oslash \mathbf{H}^\top(\mathbf{m} \otimes \mathbf{v})$
        $r \leftarrow$ solution to Global_Mass_w($\mathbf{lb}, \mathbf{ub}$)
        $\mathbf{w} \leftarrow$ Update_Vector_Global_Mass($r, \mathbf{lb}, \mathbf{ub}$)
    return $\pi = \mathbf{D_m D_v H D_w D_n}$    ▷ Relaxed transport plan
```

Algorithm 4 Relaxed Sinkhorn subroutines

```
function $\mathbf{x}$ =Update_Vector_Global_Mass($r, \mathbf{lb}, \mathbf{ub}$)
    for all $i$ do
        if $\mathbf{lb}[i] \leq r \leq \mathbf{ub}[i]$ then $\mathbf{x}[i] \leftarrow r$
        else if $\mathbf{ub}[i] < r$ then $\mathbf{x}[i] \leftarrow \mathbf{ub}[i]$
        else$\mathbf{x}[i] \leftarrow \mathbf{lb}[i]$

function $r$ = Global_Mass_v($\mathbf{lb}, \mathbf{ub}$)
    $\mathbf{x}(r)$ ←Update_Vector_Global_Mass($r, \mathbf{lb}, \mathbf{ub}$)
    $r \leftarrow$ solution to $\mathbf{m}^\top\mathbf{D_x H D_w n} = 1$

function $r$ = Global_Mass_w($\mathbf{lb}, \mathbf{ub}$)
    $\mathbf{x}(r)$ ←Update_Vector_Global_Mass($r, \mathbf{lb}, \mathbf{ub}$)
    $r \leftarrow$ solution to $\mathbf{m}^\top\mathbf{D_v H D_x n} = 1$
```

Note that each KL projection requires solving an equation to compute parameter r, which must be chosen such that the resulting transport plan has global mass equal to 1. Fortunately, the global mass is piecewise linear and non decreasing as a function of r, with a number of nodes equal to twice the number of points in each dataset. Hence, solving for r can be done by simply sorting these nodes. In the rare case that no solution exists, we set r to a large magnitude, with the appropriate sign.

Three-fold Alternating Minimizations. When bounded area distortion is allowed, our alternating minimization algorithm described in Sec. 2.3 cannot be directly extended to the case where we relax mass preservation: the denominator in our definition of transfer operators makes the transport plan optimization step more difficult, and seemingly not reducible to a convex problem. To solve this issue, we introduce two additional variables m'_i and n'_j, conceptually corresponding to respectively $\sum_j \pi_{ij}$ and $\sum_i \pi_{ij}$. Rather than enforcing strict equality, we allow slack by setting the following constraints:

$$\begin{cases} C^{-1} m'_i \leq \sum_j \pi_{ij} \leq C m'_i, \\ C^{-1} n'_j \leq \sum_i \pi_{ij} \leq C n'_j. \end{cases} \tag{10}$$

for some constant C close to 1 (set to 1.1 in our experiments). We also impose that $D^{-1} m_i \leq m'_i \leq D m_i$ and $D^{-1} n_j \leq n'_j \leq D n_j$, where D is a user-chosen bound on the allowed mass distortion. We then modify the transfer operators to:

$$\pi'_{\mathbf{X}}(\delta_{x_i}) = \frac{1}{m'_i} \sum_j \pi_{ij}\, \delta_{y_j}\,,\ \pi'_{\mathbf{Y}}(\delta_{y_j}) = \frac{1}{n'_j} \sum_i \pi_{ij}\, \delta_{x_i}\,.$$

and optimize the functional $C'(\pi, \eta)$ obtained from $C(\pi, \eta)$ in Eq. (3) by just replacing $\pi_{\mathbf{X}}$ by $\pi'_{\mathbf{X}}$ and $\pi_{\mathbf{Y}}$ by $\pi'_{\mathbf{Y}}$.

We then proceed by alternately optimizing over the three sets of variables π, (m', n'), and η. To solve for π_{ij}, we use the relaxed Sinkhorn iterations described above. Solving for m' and n' requires the optimization of a positive linear combination of inverses of their coefficients over a hypercube. This convex optimization problem is solved in $O(n \log n)$ time using an algorithm similar to the one used above to enforce global mass preservation. Finally, the update of η remains a simple weighted average as before.

4 OPTIMAL MAP SOLVER

Equipped with the algorithmic and numerical components we described above, we can now formulate an efficient and scalable solver for our inter-surface mapping approach.

4.1 Coarse-to-Fine Approach

As demonstrated by the convergence plot in Fig. 3 of the inter-curve map shown in Fig. 2, computational times for our variance-minimizing transport plan based approach may depend heavily on the size of neighborhoods we consider. While large neighborhoods make our alternating minimizations converge faster, the number of variables |**X**|+|**Y**| for two surfaces is quite large, inducing a high computational cost despite our use of fast transport plan solvers. Instead, we note that finding proper correspondences between arbitrary surfaces lends itself rather naturally to a coarse-to-fine strategy: it is quite simple to find a rough map between two simplified versions of the surfaces as the number of samples involved is low; then building a more precise map from this rough approximation becomes simpler and less prone to being stuck in a local minimum. Therefore, we operate our transport optimization on a hierarchy of approximations of the two surfaces. By starting the map optimization on a coarse sampling of the surfaces (and thus, a small number of variables) and progressively increasing resolution to refine the map, we significantly reduce the total computation time required to find a variance-minimizing map. As discussed in Sec. 3.2, using diffusion geometry for low sampling of surfaces is also particularly convenient and reliable as it leverages spectral approximations of the fine, original surfaces. This means that any instance of the positions x, y, η_x and η_y is replaced by its equivalent in diffusion space, i.e., $\hat{x}$, $\hat{y}$, $\hat{\eta}_x$ or $\hat{\eta}_y$ as detailed in Sec. 3.1. When a transport plan has been found at the finest level of our hierarchy, we then switch to a pure Euclidean stage where now the sampling is fine enough to continue refining the map using more traditional computations of tangent planes and barycenters. Throughout this hierarchical process, we also check that the map is locally well behaved to avoid degeneracies. In this section, we mostly describe the diffusion stage, but briefly mention how to adapt this process to the (simpler) Euclidean stage in Sec. 4.5. Pseudocode for our coarse-to-fine strategy is given in Alg. 5.

Algorithm 5 Overall Coarse-to-Fine Algorithm

```
function OVERALL ALGORITHM
    BUILD_HIERARCHY() // first diffusion, then Euclidean levels
    $\pi \leftarrow$ INITIALIZATION($\mathcal{S}_0$) // solve for coarsest map
    for all $\mathcal{S}_0, \mathcal{S}_1 \ldots \mathcal{S}_\ell$ do // traverse hierarchy
        $\pi \leftarrow$ ALTERNATING MINIMIZATION($\mathcal{S}_i, \pi$)
        $\pi \leftarrow$ FIX_FLIPS_AND_TWISTS($\mathcal{S}_i, \pi$)
    return $\pi$                                ▷ Mapping between X and Y
```

4.2 Constructing Hierarchy in Diffusion Space

We begin with the construction of a hierarchical sampling where the cardinality of a neighborhood, defined via a kernel-derived weighting function W_x via $\mathcal{N}_x = \{x_i \in \mathbf{X} : W_x(x_i) \geq \varepsilon\}$ (as mentioned in Sec. 3.2) remains constant as we go to finer levels in the hierarchy so as to limit computational complexity.

We leverage the smoothing properties of the time scale t in diffusion geometry (see Sec. 3.1) to construct a hierarchy of dyadic levels, one for each time scale $t_\ell = 2^{-\ell} t_0$. Starting from a large diffusion time scale $t_0 = \ln 10/\lambda_2$ [Sun et al. 2009], we build a subsampling $\mathcal{S}_0$ of the input dataset via farthest point insertion; i.e., we add one point at a time from the original pointset such that it is the farthest from the currently selected subset. We stop adding points to this level once the farthest point p just added satisfies $W_x(p) > \tau$ for the nearest neighboring point x currently in $\mathcal{S}_0$ for a value τ satisfying $1 > \tau > \varepsilon$ (we take $\tau = 0.9$ in our experiments), since it indicates that we found a dense enough sampling for that level. This approach ensures that the area contributing to the neighborhood of any point x in the level (namely, the set of points x_i satisfying $\varepsilon < W_x(x_i) < \tau$) is covered by at most a constant number of Voronoi cells, leading

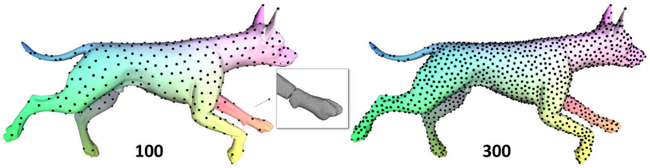


Fig. 6. **Diffusion distance estimates on thin parts.** The black dots depict the final sampling during the last, finest diffusion stage. Left: with 100 eigenvectors, sampling is insufficient to get satisfactory precision on thin parts (dog's feet, tail, ears). This may translate into twists of the mapping (middle). Right: using 300 eigenvectors significantly improves precision.

to a constant neighborhood cardinality. We then proceed with the construction of the next level using the same furthest point insertion strategy. As the time scale decreases, levels $\mathcal{S}_\ell$ become more dense until eventually, no additional points can be inserted: since only a limited number of eigenvectors are used to compute diffusion distances, our weighting functions will stop being "discerning" past a certain scale (see Fig. 6). Further levels are then computed using Euclidean distances instead, at which point local distances are almost equal to geodesic distances.

4.3 Sparsified Sinkhorn Variables

At the coarsest level of the hierarchy, we can afford to solve the full optimal transportation problem as we have considerably reduced the number of samples. However, this number steadily increases as we proceed to finer levels. In order to maintain a reasonable computational cost at higher resolutions, we limit the number of variables in the Sinkhorn iterations as follows. Consider the transport plan found so far; since our functional favors homeomorphisms, we can expect that the support of the transport plan is sparse since each point is mapped to only a small number of points as a consequence of variance minimization. Now, we consider a slightly enlarged support containing all pairs (x', y') such that there exists a pair (x, y) in the current support with $x' \in \mathcal{N}_x$ and $y' \in \mathcal{N}_y$. For the next round of Sinkhorn iterations, we limit our transport plan to this enlarged support, which limits the number of variables to instantiate (Fig. 7). Note that this process is repeated within each level of the hierarchy several times during our alternating minimization steps, so the support can gradually move to an optimal location as the centers are relocated—thus eliminating the artifacts that a naive reduction of variables in the transport plan could induce. Once the map stops evolving, we move to the next hierarchy level.

Locality. We leverage the locality of the transport map we are solving for to further accelerate the computation involved in Sinkhorn iterations. At a given level ℓ of the hierarchy, a naive implementation of Sinkhorn iterations would use a matrix $\mathbf{H}$ of size $|\mathcal{S}_\ell| \times |\mathcal{S}_\ell|$, and each iteration would have a complexity of $|\mathcal{S}_\ell|^2$. Instead, we exploit our hierarchical framework and the local instantiation of variables used by Sinkhorn iterations by storing $\mathbf{H}$ as a *sparse* matrix that records only the instantiated variables. All other matrix coefficients are set to $10 \max\{\mathrm{coef}\left(\pi_{ij}\right)\}$ to strongly penalize transport. Sinkhorn iterations are then computed efficiently by first precomputing $\sum_i \mathbf{m} \otimes \mathbf{v}[i]$ (resp., $\sum_j \mathbf{n} \otimes \mathbf{w}[j]$) in Alg. 2, then adjusting accordingly the value of $\mathbf{H}^\top(\mathbf{m} \otimes \mathbf{v})$ computed using the sparse matrix $\mathbf{H}$. As we kept the cardinality of neighborhoods constant during the construction of the hierarchy, both storage size and computational complexity involved in the optimal transport solver are *linear* in the number of samples. Fig. 8 plots the total transport cost throughout iterations while solving for an optimal map. Note that we display the transport cost normalized per level since comparing cost across levels is not meaningful: sample count, diffusion time and neighborhoods have changed. Moreover, while each level leads to a seemingly small decrease in energy, it has a significant cumulative impact across the hierarchy.

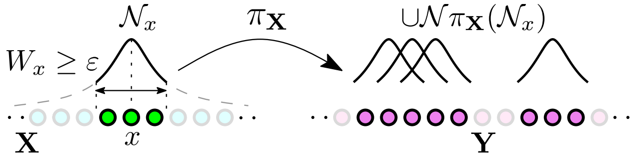


Fig. 7. **Local variable instantiation.** For a point x, we instantiate variables only in the neighborhood of the mapping of its neighborhood. Localizing samples it can be mapped to dramatically reduces the search space.

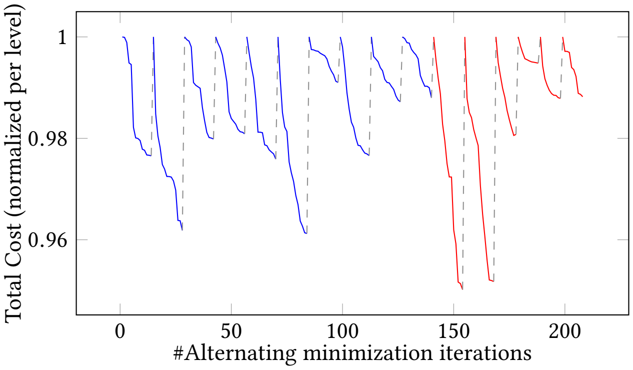


Fig. 8. **Cost reduction per level.** Decrease in total transport cost (Eq. (3), normalized per level) after every alternating minimization iteration for the example in Fig. 5. Blue indicates diffusion stage (10 levels) and red Euclidean stage (6 levels). Dashed lines represent a change of hierarchy level.

4.4 Preventing Map Flips and Twists through Ironing

While our energy was designed to induce well-behaved maps, artifacts can appear during the coarse-to-fine optimization. In particular, a few, localized discontinuities of the map may appear in the form of twists (180° rotations of the map) and/or flips (mirrored versions of the map), see Fig. 9: they are typically due to thin protrusions and/or low sampling, leading to an inadequate global map during the iterations. While more eigenvectors (Fig. 6) or more iterations with larger weighting functions at a finer scale may resolve a local flaw in the map, it would come at a very high computational cost.

We thus track potential map flaws preemptively at the end of each level of the hierarchy to fix the map early if needed. Noting that the local presence of a twist or a flip will engender a sudden change of normal directions in mapped neighborhoods, we check orientation consistency of the local map around each point $x \in \mathbf{X}$ as follows. We first project the immediate neighborhood of $\hat{x}$ on its tangent plane (computed in diffusion space, see Sec. 3.2) and build, in this 2D space, a triangulation of the convex hull of this projected neighborhood. This triangulation is then used as reference to verify the consistency of the orientation of its map by now projecting the neighborhood of $\hat{\eta}_x$ on $\mathbf{Y}$ on its own tangent plane. If any mismatch in orientation is

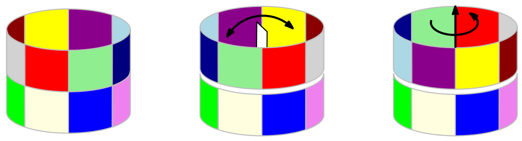


Fig. 9. **Flips and twists.** An ideal map (left); a mirrored map creating flip (middle); and 180 degree rotation of the map creating a twisted map. In practice, we found that twists rarely occur in our maps and are highly unstable as compared to flips.

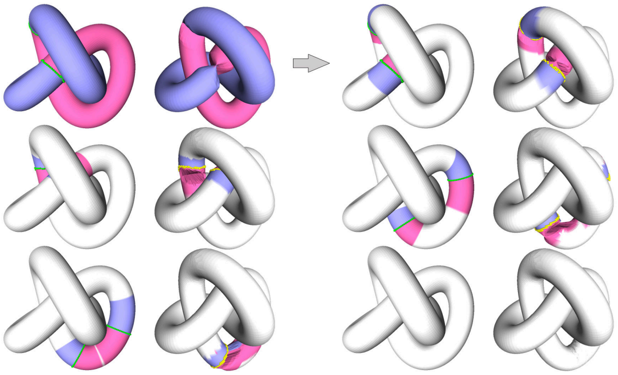


Fig. 10. **Fixing flips and twists.** Starting from a suboptimal map having a discontinuity, ironing proceeds through front propagation to remove artifacts. For each strip, blue indicates visited parts while non-visited parts are in pink. In case of multiple fronts, each front is considered independently.

found, we mark this point as inadequate. This process conservatively identifies the regions where the map is inconsistent and divides our data into separate patches delineated with badly oriented regions between them. We then systematically correct the problem areas by what is best described as an "ironing" process: we proceed with an update of the centers where we artificially increase the weights of neighborhoods where the map is good; that is, we use an asymmetric weighting function through:

$$W^*_x(x_i) = \begin{cases} W_x(x_i), & x_i \in \text{visited}, \\ \omega W_x(x_i), & \text{otherwise}, \end{cases}$$

where ω is set to 0.5 in all our experiments. This effectively overpowers any twist or flip present, and brings the solution out of the local minimum in which it was. More specifically, starting from the patch with maximum area on $\mathbf{X}$ with a correctly oriented map, we build a strip around a *front* (i.e., the delineation between already visited and non-visited points) by uniformly and geodesically thickening it. The corresponding strip is built on $\mathbf{Y}$ by geodesically thickening the image of the front on $\mathbf{X}$ proportional to the mass of the visited and non-visited parts. We then solve for a variance minimizing transport plan satisfying Eq. (1) *restricted to these two strips.* Once the centers $\hat{\eta}$ stabilize, the front is moved outward and the process is repeated until the regions marked as inadequate have been visited, eventually removing flips/twists. This ironing process is very efficient at detecting and removing artifacts of the map and its inverse even if we intentionally flip the map as a stress test, see Fig. 10.

4.5 Final Euclidean Levels

When we reach the final level of the diffusion stage, we have a fairly dense correspondence map between the two surfaces. We, however, continue with our hierarchical solver—now using the Euclidean embedding of the surfaces directly. We create finer levels in the hierarchy by reverting to a plain, unnormalized Gaussian function as our weighting function:

$$W_x(x_i) = \exp\left(-(x - x_i)^2/(2\sigma^2)\right), \tag{11}$$

where σ is initialized using the average radius r_{avg} of the neighborhoods in the last diffusion stage such that $\varepsilon = \exp(-r^2_{\text{avg}}/2\sigma^2)$. To avoid any ill-effects when σ reaches the sampling resolution of the input data, we always have a lower bound on the cardinality of the neighborhood. Furthermore, for each new Euclidean hierarchy level ℓ, σ is directly updated to $\sigma/2$. We then proceed to the repeated optimizations of the transport plan at each level as before, but now every point x and y are considered in Euclidean space directly, and the centers are computed as linear averages of these Euclidean points (which provide, at this scale, a good approximation of geodesic barycenters). It is important to point out that the Euclidean hierarchy levels do improve results quite significantly: using diffusion geometry with a limited number of eigenvectors basically ignores surface details corresponding to high frequencies. As Fig. 11 shows on a map between a sphere and a skull, details are properly adjusted once all the Euclidean levels have been treated, resulting in a sharp and variance-minimizing map as expected.

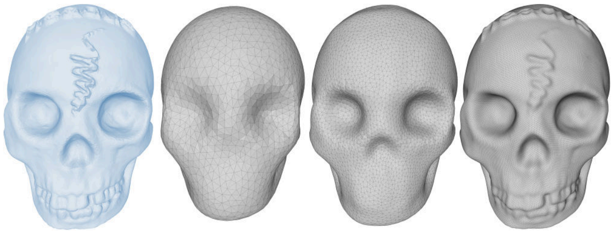


Fig. 11. **Hierarchical map optimization.** A sphere mesh (65kV) is mapped onto the input skull mesh (35kV) (left) by relocating the sphere vertices onto the skull using η, across 3 Euclidean steps of the hierarchical solve (diffusion steps are not depicted as the corresponding η's live in diffusion space).

5 EXPERIMENTS

We implemented our algorithm in C++, using the *Spectra* library [Qiu et al. 2016] to compute the eigenvectors of the Laplace operator, the *Eigen* library for linear algebra operations, CGAL [2016] for mesh data structures, the Intel Threading Building Blocks library for parallel computing and the CUDA library for GPU parallel implementation of the Sinkhorn iterations. Unless otherwise indicated, we used the first $M = 100$ eigenvectors of the Laplace operator to create diffusion coordinates. For visual evaluation of the maps we use one color per vertex and linear color interpolation per triangle. Each vertex color is set based on a smooth 3D function, sometimes modulated by a 3D axis-aligned checkerboard for enhanced clarity. In this case we further refine the reference mesh via mutual tessellation with a strip along the grid edges to improve color sharpness (called strip coloring hereafter).

Initialization. When the input models are properly rigidly aligned in space (up to a translation), minimizing only the Wasserstein-2 transport distance is sufficient to initialize η for the first level of the diffusion hierarchy. When the input models are not aligned or very different, we initialize our optimization with $\eta = 0$ and adding two to four user-defined point constraints is typically sufficient to obtain meaningful maps in these cases.

Sensitivity to parameters. While our exposition mentioned a number of parameters that the implementation depends on, their actual values only affect efficiency. For instance, our use of diffusion times being powers of two may look arbitrary: one could pick bigger increments between diffusion times which would lead to a smaller number of diffusion levels (as the number of levels is automatically found when one cannot add more samples), or smaller increments which would increase the number of levels in diffusion stage. However, the first case may trigger more alternating minimization iterations at the beginning of a new level as too sudden of a change in diffusion may induce a large change of the map; the second case may reduce the number of iterations, but results in more iterations overall as the number of levels is larger. Similarly, different values of ε and τ could be used in the diffusion stage as well. However, decreasing ε or increasing τ has an exponential impact on the cardinality of the neighborhoods. Our choice of 0.5 and 0.9 respectively has proven robust in all our examples as it roughly enforces the presence of 3-ring neighborhoods for each sample at each level. The parameter M as shown in Fig. 6, affects the quality of sampling in the diffusion stages. Picking only the first 100 eigenvectors as we do by default may not be appropriate for shapes with very thin features: it could numerically force an excessive amount of map "ironing" otherwise in the Euclidean stages of the hierarchy (as the features appear late), which an increase of M to 300 would prevent quite nicely.

Timings. Computing the eigenvectors using *Spectra* as a preprocessing step takes up to 2.5 seconds on a mesh with 5k vertices and around 9 seconds on mesh with 100k vertices for 100 eigenvectors. Defining k to be the average neighborhood cardinality and n the total number of samples, the computation cost of each Sinkhorn iteration (including building/storing $\mathbf{H}$) is $O(ck^2n)$, where c represents the transport plan "spread" (average number of points to which a point is mapped) that depends on γ. Furthermore, we parallelize our computations; consequently, our approach requires 2 to 5 mins for small (20kV) meshes, and up to 15 mins for 100kV. Note that without hierarchy and local variable instantiation, each Sinkhorn iteration would have a time and space complexity of $O(n^2)$, with an additional $O(ckn^2)$ time for building $\mathbf{H}$. Despite its overall linear complexity, our approach is compute intensive due to three nested loops: the traversal of the hierarchy, the convergence of the centers of mass and the Sinkhorn iterations.

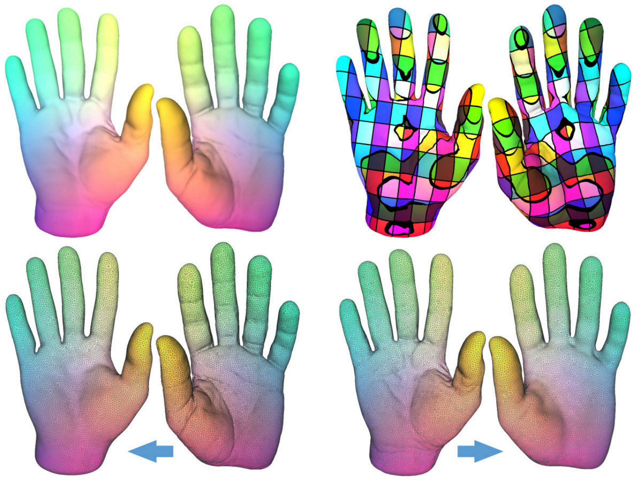


Fig. 12. **Mapping hands**. Top left: a source right hand is mapped to a target left hand based on two user-defined point-to-point correspondences. Top right: the map is depicted through a color strip. Bottom left: target mapped onto source mesh. Bottom right: source mapped onto target mesh.

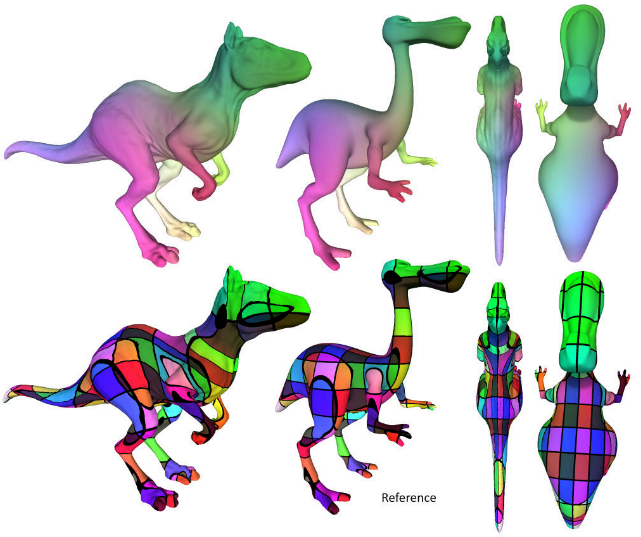


Fig. 13. **Dinos.** While the limbs, tails and heads of these two dinos greatly differ in proportion, our algorithm automatically finds a smooth and intuitive area-preserving map between the two, without any user interaction.

Inter-surface mapping examples. As our approach can find a map between arbitrary surfaces, we mostly focus on demonstrating our results on non-isometric pairs of shapes. However, the reader is invited to parse through a number of examples of near-isometric maps in Supplemental Material, where we show that our technique compares favorably to previous approaches on the simpler inter-surface mapping case.

Fig. 1 shows a baby mapped to a variety of non-isometric shapes. In this example, the user only provided two point-to-point correspondences (for the tip of the left hand/paw and the tip of the right big toe). The rest of the map, displayed through corresponding vertex colors, was found automatically through our coarse-to-fine optimization via alternating minimizations. A mass relaxation of 1.25 was allowed to avoid extreme stretching of the map.

Fig. 12 illustrates the mapping between two human hands. The hands differ in types (left vs. right) and are non-isometric. In addition, one hand extends to the wrist. Only two user-specified points are used as constraints: one to guide thumb to thumb and the other on the palm to disambiguate between front and back; the minimization algorithm then automatically finds the expected mapping. We further illustrate the mapping through cross-remeshing, i.e., by relocating the mesh vertices of the first mesh onto the center of the images (via the optimal mass transport plan) of each vertex neighborhood on the other model (and vice versa).

Fig. 13 depicts a mapping between two obviously non-isometric dinosaur models, with no area relaxation. Even if the shapes differ greatly (in particular, on the tail, limbs, and head), we can automatically find a smooth map in between them without requiring any user-prescribed correspondences.

Fig. 14 depicts a mapping between two highly non-isometric models, a man and a gorilla, with a maximum area relaxation factor set to 2 (i.e., area distortion is allowed to vary between 0.5 to 2). Four constraints were used (two hands and two feet). The resulting map is natural despite large differences in limb sizes. Note that if we further increase area relaxation to improve the map, our algorithm starts experiencing slow convergence.

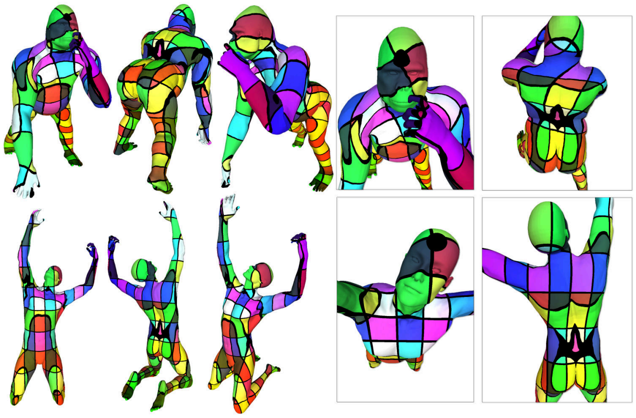


Fig. 14. **Mapping man onto gorilla.** An area distortion (up to a factor 2) is used, along with 4 constraints (one at the extremity of each limb) to map these two very different shapes.

Area preservation vs. *conformality.* Fig.15 demonstrates the behavior of our approach when the area relaxation factors vary. We compute a map between a canonical torus and a swollen torus using three different relaxation factors (from left to right : 1, 2 and 4). As expected the part mapping to the swollen region gets more mass and vice-versa. When using no relaxation, the area is exactly preserved, hence triangles from the thin part of the torus get stretched due to tangential transport. Conversely, triangles are compressed in the opposite direction on the swollen part of the torus. When area distortion is allowed, triangles remain mostly undeformed and conformality is much improved. Note that the amount of relaxation at each location is found automatically by our algorithm based on m'_i and n'_j. The user only needs to input the global mass relaxation limit (referred as D in Sec. 3.4). Also note that although the mass distribution is locally non smooth, the resulting map is smooth nonetheless since our transport stencils are a few rings wide.

Robustness. Fig. 16 depicts a mapping between two poses of a kid. While on one pose the arm merges with his chest, the resulting map is only slightly distorted. Through mapping a sphere onto three genus-1 surfaces, Fig. 17 demonstrates the robustness of our approach to topological noise and depicts the mappings when the topology differs. We modeled (1) a sphere with a small handle, (2) a sphere with a larger handle and (3) a canonical torus to test our approach. As expected the resulting maps are discontinuous but discontinuities are only present locally. No mass relaxation was used in these examples.

Evaluation. We evaluate the performance of our approach on three standard benchmark data sets: models from TOSCA [Bronstein et al. 2008], SCAPE [Anguelov et al. 2004] and SHREC'07 [Giorgi et al. 2007]. Fig.18 plots the percentage of correct correspondences against the geodesic error, averaged over all models. The units of the plots are identical to the ones used in BIM [Kim et al. 2011] to facilitate comparison. On near-isometric models from TOSCA and SCAPE our approach recovers the correct isometry in all cases and performs comparably to previous work. On SHREC models, BIM outperforms our approach (although not significantly), but our results further improve as we allow for more area relaxation: obviously, an area preserving map necessarily induces high distortion leading to bad semantic correspondences on such data. On these models we only constrained two (usually diagonally opposite) feature points. We, however, note that the curves typically shown for the SHREC models should be taken with a grain of salt: they solely measure mapping quality based on semantic feature points manually selected by a human. This is not telling much about the map: since a map is only useful in further processing task if it is smooth and valid *everywhere*, measuring performance based on a finite number of semantic points is not very informative, and provided here only for comparison purposes. In fact, we show in Fig. 19 that even for isometric shapes, BIM does not provide a useful map as obvious flaws on the ears and folds on shoulders would prevent its use in practical applications. Comparatively our maps (not driven by semantics but by local geometric criteria) are always smooth and non-degenerate.

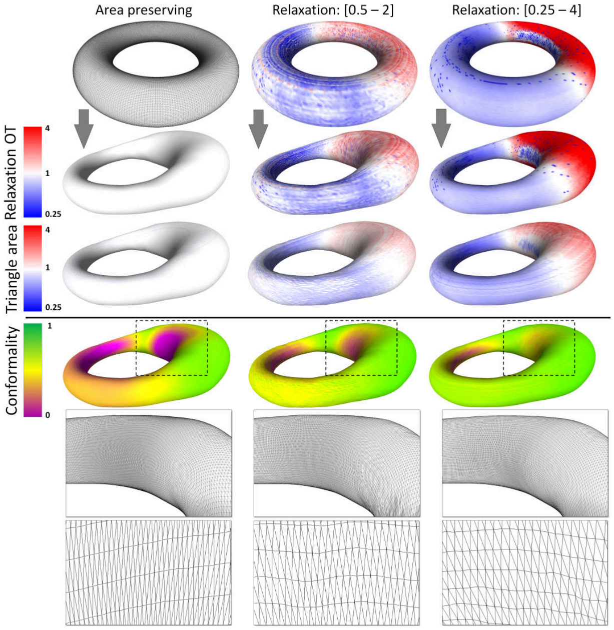


Fig. 15. **Area preservation vs. conformality.** When mass is relaxed, the transported mass $\sum_j \pi_{ij}$ changes while the original mass m_i remains identical. The slackness C helps to gradually adjust the amount of local mass-relaxation, globally bounded using the parameter D. Top: colors indicate local relaxed transported mass m'_i/m_i (blue=small, red=large). Top center: colors indicate area change per triangle (blue=shrunk, red=dilated). Bottom center: colors indicate quasi-conformal distortion per triangle (green=conformal, magenta=sheared). Larger relaxation factors allow better conformality. Bottom: closeups on the triangle meshes to highlight improved conformality.

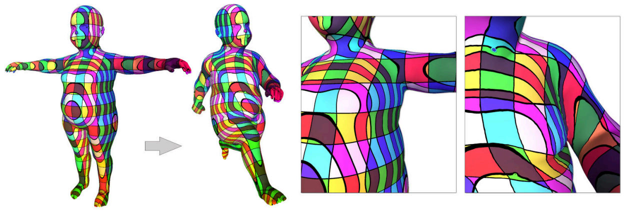


Fig. 16. **Mapping with local contact**. While the baby's arm merges with his chest on the second pose, we find a good map with only four user-selected constraints (one for each limb extremity).

Fig. 21 depicts a sample of the various SHREC models we tested and the mappings we obtained, with and without area relaxation.

Comparisons. We ran experiments to better compare our approach to BIM. It relies heavily on its initial establishment of point-to-point feature correspondences: if these correspondences fail (which happens increasingly often when shapes become non-isometric and with a wide range of feature sizes), the resulting map is nonsensical. When BIM finds good initial correspondences, their use of blended conformal maps results in visually nice maps. However, closer inspection shows severe limitations: if one relocates the vertices of the source mesh onto the target mesh via their output map, the resulting mapped mesh exhibits several artifacts as shown in Fig.19. These artifacts often make BIM maps, already limited to genus-0 surfaces, unable to help with further geometry processing operations. Our result on the same example, found without any user-specified constraints, defines a smooth and valid map everywhere. On most of the other examples we show in this paper, BIM (or the more recent non-rigid registration method of [Chen and Koltun 2015], which can only handle near-isometric models) fails to provide a usable map. More generally, even if finding sparse correspondences can be achieved by some existing methods, extending these correspondences to a continuous bijection while ensuring low distortion is no small feat (see an effort in this direction for functional correspondences in [Rodolà et al. 2015]). Instead, our approach strives to compute a dense and smooth map through a coarse to fine variational approach.

The closest approach to ours is arguably the Gromov-Wasserstein (GW) approach recently proposed by Solomon et al [2016]: they also derive a geometry-driven map through optimization as we do. Because they consider both long range and short range geodesic distances in their formulation, one could be led to believe that their method is less myopic than ours which "only" minimizes the Dirichlet energy of the map through variance minimization of increasingly small neighborhoods—albeit, in a coarse to fine fashion. However, it should be pointed out that the GW approach only finds correspondences between a small number of points (typically a few thousands in their experiments): our attempt on a denser mesh (e.g., Fig. 19 with 70k vertices) as needed for a minimally-dense map in geometry processing was already intractable given their $O(n^2)$ memory requirements and $O(n^3)$ computational complexity. In sharp contrast, both our space and time complexity are near linear in the number of samples; we were thus able to generate examples in this paper using meshes with 20k to 200k vertices. Another drawback of their methodology is that one-to-many maps can be generated as output

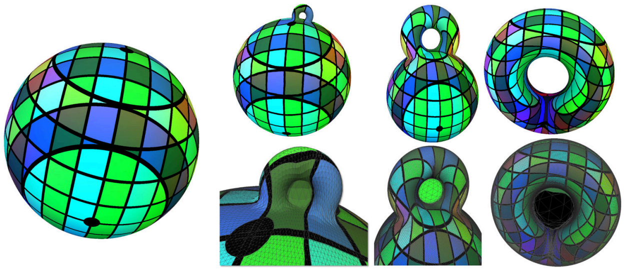


Fig. 17. **Mapping for different genus**. Middle left: a sphere with a protruding handle is mapped to a sphere (left). The map is discontinuous, but only locally. Middle right: with a bigger handle. Right: a regular torus is mapped to the sphere; the discontinuity is no longer local, but the map is continuous over half of the sphere.

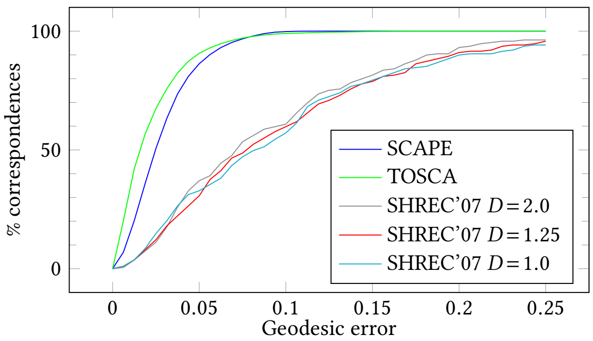


Fig. 18. **Evaluation on benchmark datasets.** We plot the percentage of correct correspondences against geodesic error, averaged over all model pairs, for various datasets. Our curves for SHREC'07 show three different bounds on allowed mass distortion.

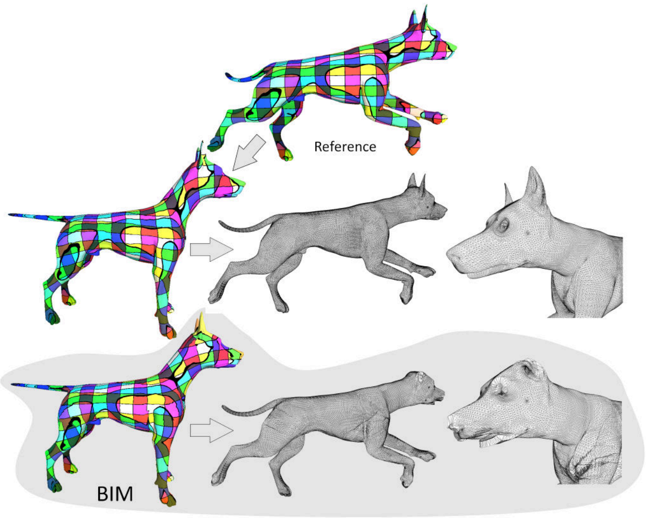


Fig. 19. **Comparison with BIM.** The dog model is mapped to another isometric pose with no user-specified constraints. Grey background: results obtained with blended intrinsic maps [Kim et al. 2011]; even the map between two isometric models is, in fact, poor—particularly on the ears and the shoulders. We also depict the meshes relocated via their respective maps. (See Supplemental Material for meshes.)

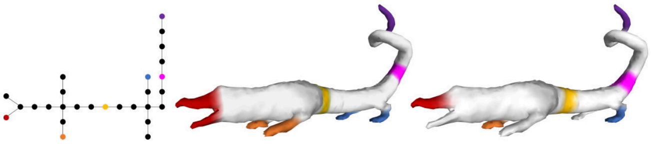


Fig. 20. **One-to-many *vs.* one-to-one maps.** Using the example of [Solomon et al. 2016] mapping a graph (left) to a surface, their approach finds correspondences to disconnected regions (center, colored disks to colored regions), while our algorithm (right) enforces locality.

(see Fig. 20), which does not meet the typical objective in geometry processing to generate sharp point-to-point maps.

Limitations. The numerics of our approach works best for uniform isotropic meshes since σ used for weighting function W is set proportionally to the vertex density to ensure that the neighborhood cardinality remains nearly constant. As a consequence, we perform recursive longest edge bisection as a preprocessing stage to obtain a more isotropic vertex distribution so as to ensure robustness. Additionally, we are also limited in the mass relaxation we can allow: relaxing the area preservation constraints through unbalanced mass transport increases the number of variables, which may slow down convergence for ratios exceeding a factor 5. In addition, if the mass is relaxed to the extent that a neighborhood maps to a single point, our algorithm will no longer be able to favor conformality locally. An obvious solution is to increase the neighborhood (by adjusting σ), but this comes again at the cost of increased computation times. Finally, while our hierarchy with power-of-two diffusion time scales have worked reliably on all the examples we tried, a better understanding of the way diffusion geometry behaves over the surface may allow for a better choice of the hierarchy, which could improve our runtimes significantly.

6 CONCLUSION

Recent progress in improving the efficiency of optimal transport solvers provides an opportunity to derive inter-surface mapping through direct energy minimization. Our paper proposes a first variational model of the sort, where transport plans are optimized to be locally sharp so as to yield homeomorphisms. We showed that our approach finds a least-stretched map with exact area preservation or bounded area distortion, and demonstrated its robustness and performance on a range of challenging pairs of models, with different and/or non-trivial topology.

One weakness of our approach is that convergence may be slower on models with a wide range of feature sizes. One direction for future work is to use Newton iterations within the nested loops to accelerate convergence of the centroids. We also would like to explore the enduring challenge of partial matching which currently remains a largely unsolved problem. Further relaxing area constraints seems to be a promising tool in this perspective. Finally, exploring other geometric priors or bounds is an interesting research direction.

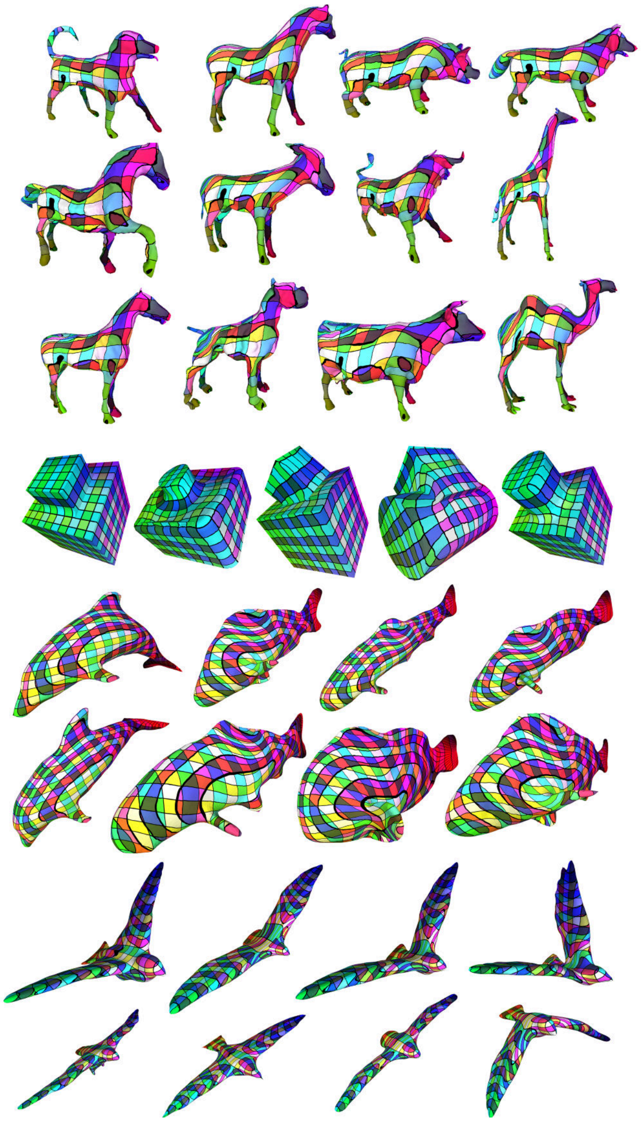


Fig. 21. **Maps from SHREC'07.** We depict mappings between a variety of SHREC models from different categories.

ACKNOWLEDGMENTS

The hands, skull and dino models are courtesy of the AIM@Shape repository; the dog, man and gorilla models are from TOSCA; the baby models from SHREC'16; Fig. 20 features a mesh from [Solomon et al. 2016], courtesy of the authors; all other models are from the SHREC'07 dataset.

This work has been funded by the European Research Council (ERC grants 257474: "IRON", 339025: "GUDHI" and 340884: "ACROSS"), NSF CCF-1655306 and the German Research Foundation (DFG, Gottfried-Wilhelm-Leibniz Programm).

We wish to thank Vladimir Kim, Justin Solomon and Qifeng Chen for providing us with their respective source codes, and the reviewers for their constructive comments. MD gratefully acknowledges the Inria International Chair program and all the members of the TITANE team for their support.

A KL PROJECTIONS

We derive the KL projection of a matrix ρ on Π_1, assuming the entries of ρ are positive. The gradient of the KL divergence to ρ at π is given by $\log \pi \oslash \rho$. By the optimality conditions, the projection π^* can be written as:

$$\log \pi^*_{ij}/\rho_{ij} = \lambda + \lambda_i + \xi_{ij},$$

where λ comes from the total mass constraint, λ_i from the row mass constraints, and ξ_{ij} from the nonnegativity condition. Parameter λ_i is zero when the mass bound is not attained, non negative if the lower mass bound is attained, and non positive if the upper mass bound is attained. Parameter ξ_{ij} is zero when π^*_{ij} is positive, and non negative otherwise. Since π^*_{ij} is positive by construction, all the ξ_{ij} are null. Let now $\phi_i = e^{\lambda} e^{\lambda_i}$ and $\phi = e^{\lambda}$. Using these variables, the optimality conditions read:

$$\begin{cases} \phi_i = \phi & \text{if } \sum_j \rho_{ij}\phi_i \in (\alpha_i m_i, \beta_i m_i) \\ \phi_i \leq \phi & \text{if } \sum_j \rho_{ij}\phi_i = \beta_i m_i \\ \phi_i \geq \phi & \text{if } \sum_j \rho_{ij}\phi_i = \alpha_i m_i \end{cases}$$

So ϕ_i is a function of ϕ: $\phi_i = f_i(\phi)$, where f_i is the closest projection on the interval $[\alpha_i m_i / \sum_j \rho_{ij}, \beta_i m_i / \sum_j \rho_{ij}]$, which is a nondecreasing piecewise linear function with two nodes. In order to determine the correct value for ϕ, it suffices to solve the equation $\sum_{ij} \rho_{ij} f_i(\phi) = 1$, so that the global mass constraint is satisfied. Taking into account the change of variables due to the area factors **m** and **n** yields the algorithm described in Section 3.4.

REFERENCES

Yonathan Aflalo, Ron Kimmel, and Michael Zibulevsky. 2013. Conformal Mapping with as Uniform as Possible Conformal Factor. *SIAM J. Imaging Sciences* 6, 1 (2013), 78–101.

Noam Aigerman and Yaron Lipman. 2015. Orbifold Tutte embeddings. *ACM Trans. Graph.* 34, 6 (2015), Art. 190.

Noam Aigerman, Roi Poranne, and Yaron Lipman. 2014. Lifted bijections for low distortion surface mappings. *ACM Trans. Graph* 33, 4 (2014), Art. 69.

Noam Aigerman, Roi Poranne, and Yaron Lipman. 2015. Seamless surface mappings. *ACM Trans. Graph.* 34, 4 (2015), Art. 72.

Dragomir Anguelov, Praveen Srinivasan, Hoi-Cheung Pang, Daphne Koller, Sebastian Thrun, and James Davis. 2004. The Correlated Correspondence Algorithm for Unsupervised Registration of Nonrigid Surfaces. In *Proc. of Neural Information Processing Systems (NIPS)*. 33–40.

Arul Asirvatham, Emil Praun, and Hugues Hoppe. 2005. Consistent Spherical Parameterization. In *Int. Conf. Computational Science*. Springer, 265–272.

Heinz H. Bauschke and Adrian S. Lewis. 1998. Dykstra's algorithm with Bregman projections: a convergence proof. *Optimization* 48 (1998), 409–427.

Jean-David Benamou, Guillaume Carlier, Marco Cuturi, Luca Nenna, and Gabriel Peyré. 2015. Iterative Bregman Projections for Regularized Transportation Problems. *SIAM J. Scient. Comp.* 37, 2 (2015).

Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael M. Bronstein. 2016. Learning Shape Correspondence with Anisotropic Convolutional Neural Networks. In *Proceedings of Neural Information Processing Systems (NIPS)*.

Alexander Bronstein, Michael Bronstein, and Ron Kimmel. 2008. *Numerical Geometry of Non-Rigid Shapes* (1 ed.). Springer.

Alexander M. Bronstein, Michael M. Bronstein, and Ron Kimmel. 2006. Generalized multidimensional scaling: A framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences* 103, 5 (2006), 1168–1172.

CGAL Project. 2016. *CGAL User and Reference Manual* (4.8 ed.). CGAL Editorial Board. http://doc.cgal.org/4.8/Manual/packages.html

Qifeng Chen and Vladlen Koltun. 2015. Robust Nonrigid Registration by Convex Optimization. In *Proceedings of ICCV*. 2039–2047.

Lenaic Chizat, Gabriel Peyré, Bernhard Schmitzer, and François-Xavier Vialard. 2015. Unbalanced Optimal Transport: Geometry and Kantorovich Formulation. (2015). http://arxiv.org/abs/1508.05216

Lenaic Chizat, Gabriel Peyré, Bernhard Schmitzer, and François-Xavier Vialard. 2016. Scaling Algorithms for Unbalanced Transport Problems. (2016). https://arxiv.org/abs/1607.05816

Ronald R. Coifman, Stéphane Lafon, Ann B. Lee, Mauro Maggioni, F Frederick J. Warner, and Steven W. Zucker. 2005. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. In *National Academy of Sciences*, Vol. 102(21). 7426–7431.

Etienne Corman, Maks Ovsjanikov, and Antonin Chambolle. 2015. Continuous Matching via Vector Field Flow. *Comput. Graph. Forum* 34, 5 (2015), 129–139.

Marco Cuturi. 2013. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. *Adv. Neural Inf. Process. Syst.* 26 (2013), 2292–2300.

Charlie Frogner, Chiyuan Zhang, Hossein Mobahi, Mauricio Araya-Polo, and Tomaso Poggio. 2015. Learning with a Wasserstein Loss. In *Proceedings of Neural Information Processing Systems (NIPS)*. 2053–2061.

S. Gallot, G. Besson, and P. Bérard. 1994. Embedding Riemannian Manifolds by Their Heat Kernel. *Geom. Funct. Anal.* 4, 4 (1994), 373–398.

Daniela Giorgi, Silvia Biasotti, and Laura Paraboschi. 2007. Shape retrieval contest 2007: Watertight models track. *SHREC competition* 8 (2007).

Vladimir G. Kim, Yaron Lipman, and Thomas A. Funkhouser. 2011. Blended intrinsic maps. *ACM Trans. Graph* 30, 4 (2011), Art. 79.

Vladislav Kraevoy and Alla Sheffer. 2004. Cross-parameterization and Compatible Remeshing of 3D Models. In *Proceedings of ACM SIGGRAPH*, Vol. 23(3). 861–869.

Yaron Lipman. 2012. Bounded distortion mapping spaces for triangular meshes. *ACM Trans. Graph.* 31, 4 (2012), Art. 108.

Facundo Mémoli. 2011. Gromov-Wasserstein Distances and the Metric Approach to Object Matching. *Foundations of Computational Mathematics* 11, 4 (2011), 417–487.

Ashish Myles and Denis Zorin. 2013. Controlled-distortion constrained global parametrization. *ACM Trans. Graph.* 32, 4 (2013), 105:1–105:14.

Boaz Nadler, Stephane Lafon, Ioannis Kevrekidis, and Ronald R. Coifman. 2005. Diffusion Maps, Spectral Clustering and Eigenfunctions of Fokker-Planck Operators. *Appl. Comput. Harmon. Anal.* 21, 1 (2005), 113 – 127.

Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas J. Guibas. 2012. Functional maps: a flexible representation of maps between shapes. *ACM Trans. Graph.* 31, 4 (2012), Art. 30.

Yixuan Qiu, Gael Guennebaud, and Jitse Niesen. 2016. Spectra: A header-only C++ library for large scale eigenvalue problems. (2016). http://yixuan.cos.name/spectra/

Anand Rangarajan, Alan L. Yuille, and Eric Mjolsness. 1999. Convergence Properties of the Softassign Quadratic Assignment Algorithm. *Neural Computation* 11, 6 (1999), 1455–1474.

Emanuele Rodolà, Michael Moeller, and Daniel Cremers. 2015. Point-wise Map Recovery and Refinement from Functional Correspondence. In *Proceedings of VMV*. 25–32.

Bernhard Schmitzer. 2016. Scaling Algorithms for Unbalanced Transport Problems. (2016). https://arxiv.org/abs/1610.06519

John Schreiner, Arul Asirvatham, Emil Praun, and Hugues Hoppe. 2004. Inter-surface Mapping. In *Proceedings of ACM SIGGRAPH*. 870–877.

Alon Shtern and Ron Kimmel. 2015. Spectral gradient fields embedding for nonrigid shape matching. *Computer Vision and Image Understanding* 140 (2015), 21–29.

Richard Sinkhorn. 1964. A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices. *Annals Math. Statist.* 35 (1964).

Justin Solomon, Fernando de Goes, Gabriel Peyré, Marco Cuturi, Adrian Butscher, Andy Nguyen, Tao Du, and Leonidas Guibas. 2015. Convolutional Wasserstein Distances: Efficient Optimal Transportation on Geometric Domains. *ACM Trans. Graph.* 34, 4 (2015).

Justin Solomon, Leonidas J. Guibas, and Adrian Butscher. 2013. Dirichlet Energy for Analysis and Synthesis of Soft Maps. *Comp. Graph. Forum* 32, 5 (2013), 197–206.

Justin Solomon, Andy Nguyen, Adrian Butscher, Mirela Ben-Chen, and Leonidas J. Guibas. 2012. Soft Maps Between Surfaces. *Comp. Graph. Forum* 31, 5 (2012), 1617–1626.

Justin Solomon, Gabriel Peyré, Vladimir G. Kim, and Suvrit Sra. 2016. Entropic Metric Alignment for Correspondence Problems. *ACM Trans. Graph.* 35, 4 (2016), Art. 72.

Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. 2009. A Concise and Provably Informative Multi-scale Signature Based on Heat Diffusion. In *Symposium on Geometry Processing*. 1383–1392.

Oliver van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. 2011. A Survey on Shape Correspondence. *Comp. Graph. Forum* 30, 6 (2011), 1681–1707.

Matthias Vestner, Roee Litman, Emanuele Rodolà, Alex Bronstein, and Daniel Cremers. 2017. Product Manifold Filter: Non-rigid shape correspondence via kernel density estimation in the product space. In *Proceedings of CVPR*.