Eurographics Symposium on Geometry Processing 2018
T. Ju and A. Vaxman
(Guest Editors)

Volume 37 (2018), Number 5

Interactive Curve Constrained Functional Maps

A. Gehre[1,2] M. Bronstein[3] L. Kobbelt[2] J. Solomon[1]

[1]Massachusetts Institute of Technology
[2]RWTH Aachen University
[3]Imperial College London

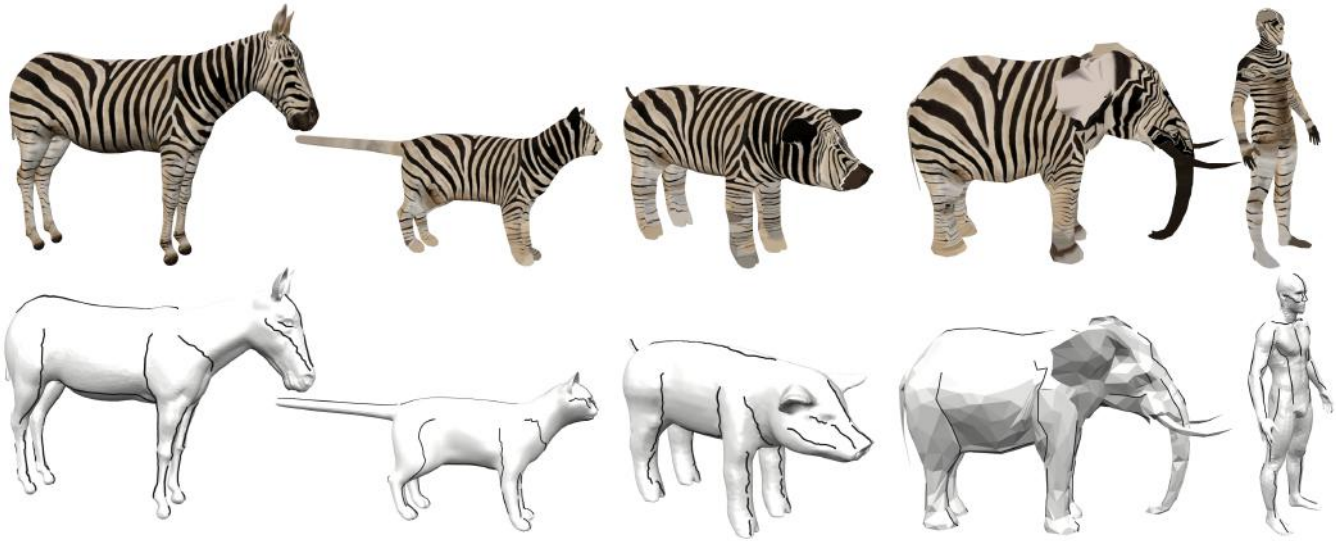


Figure 1: *We present an interactive approach to functional map computation that finds smooth maps between semantically similar but geometrically different shapes. The user prescribes corresponding feature curves, which can be computed automatically from their endpoints (bottom row). We incorporate feature curve preservation into functional map computation and show that descriptors are no longer required. We subsequently obtain point correspondences from the curve-based functional map using [ESBC18] (top row).*

Abstract

Functional maps have gained popularity as a versatile framework for representing intrinsic correspondence between 3D shapes using algebraic machinery. A key ingredient for this framework is the ability to find pairs of corresponding functions (typically, feature descriptors) across the shapes. This is a challenging problem on its own, and when the shapes are strongly non-isometric, nearly impossible to solve automatically. In this paper, we use feature curve correspondences to provide flexible abstractions of semantically similar parts of non-isometric shapes. We design a user interface implementing an interactive process for constructing shape correspondence, allowing the user to update the functional map at interactive rates by introducing feature curve correspondences. We add feature curve preservation constraints to the functional map framework and propose an efficient numerical method to optimize the map with immediate feedback. Experimental results show that our approach establishes correspondences between geometrically diverse shapes with just a few clicks.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computer graphics—Computational Geometry and Object Modeling

1. Introduction

Three-dimensional shape correspondence is essential to geometry processing, serving as a basic component of algorithms for statistical shape analysis, texture transfer, shape interpolation, and other tasks. Correspondence tools enable transfer of information from one shape to another, usually guided by geometric and semantic

cues suggesting which features on one shape should be matched to features on another.

Solving the correspondence problem directly for a point-to-point map, however, is difficult to pose and often leads to computationally-intensive algorithms. Functional maps [OBCS*12] provide an efficient alternative structure to relate shapes. Rather than matching points directly, functional maps transport *functions*, represented in a low-dimensional basis on the two shapes. The operators transporting functions from one shape to another are linear, leading to fast algorithms for computing functional maps using linear algebra and convex optimization.

Most functional map algorithms rely strongly on the ability to provide pairs of corresponding *descriptor* functions over the two shapes. For this reason, they are based on the assumption that functions e.g. local point signatures, which are computed independently on both shapes, are comparable. Reliance on high-quality descriptors becomes a key limitation in difficult correspondence settings: functional maps are effective for shapes for which it is straightforward to extract pairs of matching descriptors—e.g. nearly-isometric surfaces—and otherwise can become unreliable.

When shapes vary significantly, the assumption that point signatures or segmentations can be matched breaks down. In this regime, automatically-computed descriptors can be misleading for map computation and can give unsatisfying results. Finding a matching set of descriptors for a given pair of shapes becomes tedious and nearly as difficult as the correspondence problem itself. Algorithms designed to handle this scenario often iteratively recompute the map, adjust pointwise signatures, or add constraints that make the problem difficult to solve efficiently. Recent attempts at constructing descriptors with deep learning [LRR*17] show promising results but require a training set of shapes with known correspondences, which is not always available.

In this paper, we compute functional maps between extremely non-isometric shapes by incorporating *curve constraints* in an interactive feedback loop; these constraints disambiguate poor or misleading geometric cues without incurring significant computational cost. Our technique is based on the observation that semantically similar shapes can be abstracted by feature curve networks, in which corresponding feature curves describe related parts. This representation is extremely flexible, since the shape of the curve along the geometry as well as its length suggest how the shapes relate without imposing hard isolated point constraints.

We develop a graphical user interface, allowing to interactively and iteratively build and refine the correspondence between a pair of shapes. In our system, the user provides sparse input indicating the shapes or start- and end-points of matching feature curves along two surfaces. This input is simple and can be provided by non-experts; furthermore, the intermediate connectivity of the curves can be constructed automatically. Behind the scenes, we incorporate feature curve constraints into the functional map optimization problem and are able to update the map using an efficient iterative algorithm to provide *interactive* feedback to the user. We demonstrate that high-quality functional maps can be created using only feature curve constraints (resulting from just a few user clicks), removing the reliance on computation of point signatures.

1.1. Contributions

Our contributions can be summarized as follows:

- We present an interactive method to compute functional maps based on feature curve correspondences.
- For the computation of the functional maps, we propose descriptor-free feature curve constraints and demonstrate that they are sufficient to extract a high-quality map.
- We present a conjugate gradients optimization procedure to update the functional map efficiently.
- To show the effectiveness of our method, we use our maps to compute point correspondences and evaluate quality by performing correspondence accuracy tests.

2. Related Work

There is a vast amount of research on nonrigid shape matching; see e.g. [VKZHCO11] for a broad survey. Here, we focus on those techniques most comparable or relevant to our approach.

Point signatures. A classical approach for finding correspondences between shapes involves constructing a local descriptor (point signature) at each point and then matching points with the most similar descriptors. Also region correspondences [GSTOG16] can be found by analyzing the rank of descriptor values. Somewhat ad hoc but often effective techniques for designing shape descriptors include constructions like collecting histograms of orientations or curvature values within a fixed radius around points of interest; see [STDS14, LWWS15] for recent examples.

In applications involving deformable shapes, many modern techniques are designed to be invariant to isometric deformations. A large variety of isometry-invariant descriptors is based on intrinsic constructions such as geodesic distances [LGB*13], local polar coordinates [KBLB12], or eigenvectors and eigenvalues of the Laplace-Beltrami operator [Rus07, SOG09, BK10, ASC11, OMMG10]. Among spectral techniques, the *heat kernel signature* (HKS) [SOG09], measuring heat auto-diffusion of each point of the shape, and the wave kernel signature (WKS) measuring the evolution of a quantum particle on the surface [ASC11] are especially popular. Modifications of these descriptors incorporate scale-invariance [BK10], learning [LB14], and anisotropy [BMR*16].

Most recently, intrinsic versions of convolutional neural networks (CNNs) automatically learn descriptors from collections of shapes with annotated correspondences [MBBV15, BMRB16]. Such methods can be superior to handcrafted features but require a large training set of corresponding shapes.

Curve networks. In this paper, we use *feature curve networks* (FCNs) to provide hints about significant parts of a shape and its underlying structure. Feature curves are relatively well-studied in computer graphics and geometry processing with a number of approaches for their extraction. Classical approaches include finding curvature extrema [WB01], tracing ridges and ravines [HPW05] and crest lines [YBS05], and constructing the feature skeleton [WG09]. Denoising methods from FCNs were proposed in [GLK18]. For an in-depth overview of FCN detection methods, we refer the reader to [LZH*07, DVVR07].

Functional maps. Traditional methods for shape correspondence rely on a point-to-point map representation. Working with such representation often leads to algorithms with high computational complexity. Instead, *functional maps* [OBCS*12] model correspondences as operators between functional spaces. These operators can be efficiently expressed with respect to bases for function spaces on the shapes; typically, the orthogonal Laplace–Beltrami eigenbases are used. In the simplest setting, functional map computation boils down to a least-squares problem for the map representation coefficients that best preserve a collection of descriptors [OBCS*12]. These descriptors include point signatures as well as segment or point correspondences encoded e.g. using wave kernel maps with fixed source points/regions.

Functional maps can be improved by introducing various priors. [KBB*13, EKB*15] use joint diagonalization to develop compatible bases on the two surfaces in the near-isometric case. [PBB*13] use ideas from sparse recovery to improve the functional mapping pipeline and associated bases. [ERGB16] use coupled functional maps to obtain consistent mappings from the source to the target shape and vice versa. Also, partial functional maps [RCB*17] to overcome the problem of missing data in case a full scan of the source or target is not provided. [HWG14] extend to collections of more than two shapes, computing consistent maps in shape collections. Finally, descriptor preservation can be improved by promoting commutativity with pointwise multiplication operators [NO17] and by approximately preserving of products of functions [NMR*18].

The applicability of the framework is heavily restricted by the need for compatible descriptors across shapes, which practically limits the technique to nearly-isometric shapes. When the geometry differs and the point signatures do not reliably match, it is hard to find accurate maps automatically.

While map computation is easier in the functional maps framework (a linear system as opposed to a combinatorial problem), the resulting map does not take points to points. Conversion of functional maps into point-wise maps while preserving geometric properties like bijectivity or smoothness is a challenging problem that continues to be studied. In [OBCS*12], this conversion is done by searching nearest neighbors or by an ICP-type rigid alignment in the target shape basis. Recent alternatives include methods for functional maps denoising [EBC17], kernel density estimation on the product of the two shapes [VLR*17], and alternating diffusion/sharpening [B*17]. Finally, recent work [ESBC18] achieves the best results in our experiments by optimizing for a nearly-harmonic map which is reversible with geodesic consistency.

3. Background

Given a pair of shapes (modeled as surfaces M and N), correspondence algorithms seek a map $\tau : M \to N$ that preserves some structure, e.g. a pointwise geometric quantity like curvature or a segmentation into semantically-meaningful parts. Additional properties can be imposed on τ, e.g. bijectivity or smoothness. Two classes of relationships between M and N distinguish settings of the correspondence problem:

Geometric similarity: When M and N are nearly isometric, the desired correspondence preserves the metric as well as intrinsic structures like geodesic distances and Laplacian spectra. The isometry assumption is used widely in nonrigid correspondence, since it holds approximately for inelastic and articulated deformation. Correspondence quality in this setting can be quantified by measuring intrinsic distortion, giving a natural objective function for correspondence algorithms.

Semantic similarity: More generally, shapes are not necessarily isometric but rather have semantically corresponding structures that should be preserved by the map τ. For example, dogs and cats are quadrupeds; a meaningful correspondence would map legs to legs, the tail to tail, and so on. This notion of similarity is less precise and can be hard to quantify; correspondence is often guided by aesthetic considerations and may need manual user annotation.

Functional maps. The key idea of functional maps is to replace pointwise correspondences $\tau : M \to N$ with operators $T : L^2(M) \to L^2(N)$. The former is a particular setting of the latter when τ is bijective, since one can define $T_\tau f = f \circ \tau^{-1}$. T, however, is of interest independently, admitting a matrix representation with respect to (usually orthogonal) bases $\{\phi_i\}_{i\geq 1} \subseteq L^2(M)$ and $\{\psi_j\}_{j\geq 1} \subseteq L^2(N)$ taking the coefficients of a function f in the ϕ_i basis on M to those of Tf in the ψ_j basis on N.

It is common to use the eigenfunctions of the *Laplace–Beltrami operator* $\Delta_M \phi_i = \lambda_i \phi_i$ as an orthogonal basis for $L^2(M)$ (respectively, $L^2(N)$); here, $0 = \lambda_1 \leq \lambda_2 \leq \dots$ denote the corresponding eigenvalues. After truncating the correspondence matrix to the first k coefficients in the two bases, the functional map is represented by a $k \times k$ matrix $\mathbf{C} = (c_{ij})$. Unless otherwise noted, henceforth we will assume functional maps are written in the Laplace–Beltrami basis.

Given a set of $q \geq k$ corresponding functions $f_1, \dots, f_q \in L^2(M)$ and $g_1, \dots, g_q \in L^2(N)$ such that $Tf_i \approx g_i$, it is possible to recover $\mathbf{C}$ by solving the linear system of equations

$$\mathbf{C}\hat{\mathbf{F}} = \hat{\mathbf{G}},$$

where $\hat{\mathbf{F}} = \left(\langle f_i, \phi_j \rangle_{L^2(M)}\right)^\top$ and $\hat{\mathbf{G}} = \left(\langle g_i, \psi_j \rangle_{L^2(N)}\right)^\top$ are $k \times q$ matrices containing as columns the coefficients of the functions f_i and g_i in the Laplace–Beltrami eigenbases. If the map is area-preserving, it can be shown to commute with the Laplace–Beltrami operator, resulting in additional constraints $\mathbf{C}\mathbf{\Lambda}_M = \mathbf{\Lambda}_N \mathbf{C}$ [OBCS*12], with $\mathbf{\Lambda} = diag(\lambda_1, \dots, \lambda_k)$.

In the discrete setting, manifolds M and N are discretized as triangular meshes with m and n vertices v_i, respectively. Functions on M can be identified with m-dimensional column vectors $\mathbf{f} = (f(v_1), \dots, f(v_m))^\top$, and inner products are written as $\mathbf{f}^\top \mathbf{A}_M \mathbf{g}$, where $\mathbf{A}_M = \mathrm{diag}(a_1, \dots, a_m)$ is the diagonal matrix of local area elements. The (truncated) Laplace–Beltrami operators assume the form $\mathbf{\Delta}_M \approx \mathbf{\Phi}\mathbf{\Lambda}_M\mathbf{\Phi}^\top\mathbf{A}_M$ and $\mathbf{\Delta}_N \approx \mathbf{\Psi}\mathbf{\Lambda}_N\mathbf{\Psi}^\top\mathbf{A}_N$, where $\mathbf{\Phi} = (\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_k)$ and $\mathbf{\Psi} = (\boldsymbol{\psi}_1, \dots, \boldsymbol{\psi}_k)$ contain the eigenvectors. Fourier decomposition can be written in matrix-vector form as $\hat{\mathbf{F}} = \mathbf{\Phi}^\top \mathbf{A}_N \mathbf{F}$ and $\hat{\mathbf{G}} = \mathbf{\Psi}^\top \mathbf{A}_M \mathbf{G}$, where matrices $\mathbf{F}$ and $\mathbf{G}$ of size $m \times q$ and $n \times q$, respectively contain descriptors as columns. The functional map is discretized as an $n \times m$ matrix $\mathbf{T}$, and its representation in the truncated basis can be considered as a k-rank approximation: $\mathbf{T} \approx \mathbf{\Psi}\mathbf{C}\mathbf{\Phi}^\top\mathbf{A}_M$.

The crucial part of of the functional maps framework is the requirement to provide pairs of corresponding functions. Automatically constructing such functions, by computing local descriptors, is often practically limited to the near-isometric setting. When these strong geometric assumptions are violated, it is difficult to detect similar structures automatically. In this paper, we instead leverage sparse user guidance to provide semantic information; we will demonstrate cases in which a meaningful smooth map can be extracted between shapes even if their intrinsic geometric structures differ considerably.

4. Interactive Functional Maps

We propose incorporating human interaction into the correspondence process, allowing for efficient *interactive* functional map design. Several desiderata inform our design:

- INPUT: The user should be provided with a simple, fast, and effective method to input semantic information. For this purpose, we use ordered curve networks that are computed automatically from user-provided endpoints; effectively, this reduces the user input to a few mouse clicks.
- OUTPUT: The feedback to the user should be given in a way that he or she can assess the map quality and edit the set of feature curves accordingly.
- SPEED: The functional map computation and display of the feedback to the user must be provided at interactive rates.

Curve networks. The input provided by the user is in the form of a set of q corresponding curves on M and N, which we denote by $\gamma_1,\dots,\gamma_q$ and $\eta_1,\dots,\eta_q$, respectively. Since drawing curves on surfaces is challenging, the user is requested to provide only the endpoints of the curves, which are connected automatically.We use Dijkstra's shortest path algorithm along mesh edges to connect the endpoints, with two options for weights:

- GEODESIC DISTANCE: The weight for an edge $e=(p_1,p_2)$ is its length in $\mathbb{R}^3$, providing an approximation of a geodesic curve.
- ANISOTROPIC DISTANCE: The weight for an edge e is biased to prefer ridge and valley curves by incorporating the curvature tensor as in [CHK13].

For simplicity, our segments coincide with triangle edges. Our method, however, extends easily to the general case, simply requiring a generalized definition of the restriction operators below; we found that the increased precision from this more complex computation was negligible.

Functional maps with curve constraints. Let $\gamma\colon[0,1]\to M$ and $\eta\colon[0,1]\to N$ be two corresponding curves on M and N. Denote by $P_\gamma^M : L^2(M)\to L^2([0,1])$ the *restriction operator* taking a function on M and outputting its values along the curve γ, defined as $P_\gamma^M f = f\circ\gamma$. Then, the fact that γ and η correspond can be expressed as

$$P_\gamma^M f = P_\eta^N T f, \tag{1}$$

for any function $f\in L^2(M)$. In particular, applying (1) to the basis functions themselves, we get

$$\underbrace{P_\gamma^M \phi_i}_{\bar\phi_i^\gamma} = P_\eta^N \sum_{\ell=1}^{k} c_{\ell i}\psi_\ell = \sum_{\ell=1}^{k} c_{\ell i}\underbrace{P_\eta^N\psi_\ell}_{\bar\psi_\ell^\eta}, \tag{2}$$

which can be thought of as the restriction of the functional map to the curve.

In the discrete setting, we assume that the curves are uniformly sampled at s samples. Then, the restriction operators can be represented as sparse $s\times m$ and $s\times n$ matrices $\mathbf{P}_\gamma^M$ and $\mathbf{P}_\eta^N$, respectively. Equation (2) can be rewritten in matrix form as

$$\mathbf{P}_\gamma^M\mathbf{\Phi} = \mathbf{P}_\eta^N\mathbf{\Psi}\mathbf{C} \tag{3}$$

$$\bar{\mathbf{\Phi}}^\gamma = \bar{\mathbf{\Psi}}^\eta\mathbf{C}, \tag{4}$$

where $\bar{\mathbf{\Phi}}^\gamma$ and $\bar{\mathbf{\Psi}}^\eta$ are $s\times k$ matrices representing the basis functions restricted to the respective curves.

Given a collection of q corresponding curves $\gamma_1,\dots,\gamma_q$ and $\eta_1,\dots,\eta_q$, we apply the constraint (3) in the form of a penalty together with the commutativity penalty, to find the functional map matrix $\mathbf{C}$ minimizing the energy

$$\mathcal{E}(\mathbf{C}) = \alpha\|\mathbf{C}\mathbf{\Lambda}_M - \mathbf{\Lambda}_N\mathbf{C}\|_{\mathrm{F}}^2 + \sum_{\ell=1}^{q}\|\bar{\mathbf{\Psi}}_\ell\mathbf{C} - \bar{\mathbf{\Phi}}_\ell\|_{\mathrm{F}}^2, \tag{5}$$

where $\bar{\mathbf{\Phi}}_\ell = \bar{\mathbf{\Phi}}^{\gamma_\ell}$ and $\bar{\mathbf{\Psi}}_\ell = \bar{\mathbf{\Psi}}^{\eta_\ell}$.

Iterative update. To give immediate feedback to the user after adding, deleting, or updating a curve, we present an efficient technique to update the functional map with a new objective term in (5). Each user input updates the second (data) term of the energy $\mathcal{E}(\mathbf{C})$. Since this is a least-squares problem, implicitly we need to solve a linear system $\mathbf{H}\mathbf{x}=\mathbf{b}$ in each iteration (here $\mathbf{x}$ is the k^2-dimensional vectorized form of the $k\times k$ functional map matrix $\mathbf{C}$), which has a complexity of $\mathcal{O}(k^6)$ for a dense $\mathbf{H}\in\mathbb{R}^{k^2\times k^2}$.

Since we can reuse previous map estimates as initial guesses, in our implementation iterative solvers converge quickly and lead to a performance improvement over direct solvers. Our linear system is guaranteed to be positive-definite since it comes from a least-squares problem. Hence, we formulate an efficient optimization procedure adapting the *conjugate gradients* (CG) algorithm for positive definite systems.

The conjugate gradients algorithm finds minima of convex functions of the form $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top\mathbf{H}\mathbf{x} - \mathbf{x}^\top\mathbf{b} + c$ [HS52,Sol15]; the first-order optimality condition is the linear system $\nabla f(\mathbf{x}) = \mathbf{H}\mathbf{x}-\mathbf{b} = \mathbf{0}$. It requires multiplication by the positive-definite matrix $\mathbf{H}$ (the linear part of ∇f) as well as access to the elements of $\mathbf{b}$ (the constant part of ∇f); it does not require direct access to elements of $\mathbf{H}$.

In our case, computing the matrix derivative of $\mathcal{E}(\mathbf{C})$ with respect to $\mathbf{C}$ yields the gradient in the canonical form

$$\nabla_{\mathbf{C}}\mathcal{E}(\mathbf{C}) = \underbrace{-2\sum_{\ell=1}^{q}\bar{\mathbf{\Psi}}_\ell^\top\bar{\mathbf{\Phi}}_\ell}_{\mathbf{B}} + \underbrace{2\alpha\left(\mathbf{\Lambda}_N^2\mathbf{C} - \mathbf{\Lambda}_M\mathbf{C}\mathbf{\Lambda}_N - \mathbf{\Lambda}_N\mathbf{C}\mathbf{\Lambda}_M + \mathbf{C}\mathbf{\Lambda}_M^2\right) + 2\sum_{\ell=1}^{q}\bar{\mathbf{\Psi}}_\ell^\top\bar{\mathbf{\Psi}}_\ell\mathbf{C}}_{\mathbf{H}(\mathbf{C})}$$

where $\mathbf{H}(\mathbf{C})$ and $\mathbf{B}$ denote the linear and constant terms, respec-

Figure 2: *Iterations in the user interface. From left to right the user adds more feature curve correspondences to improve the quality of the functional map. The texture is updated after new curves are included by transporting per-vertex texture coordinates with the functional map.*

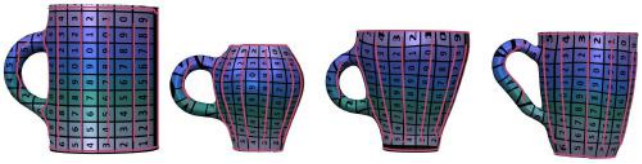


Figure 3: *Corresponding feature curves (red) on a set of cups with texture coordinates mapped by the functional map. The left object is the source shape while the other shapes are target meshes.*

Algorithm 1 Interactive Functional Maps

function INTERACTIVE FUNCTIONAL MAPS($\mathbf{\Lambda}_M$, $\mathbf{\Lambda}_N$, α)
  $\mathbf{T}_1 \leftarrow 2\alpha\mathbf{\Lambda}_N^2$
  $\mathbf{T}_2 \leftarrow 2\alpha\mathbf{\Lambda}_M^2$
  $\mathbf{B} \leftarrow \mathbf{0}$
  initialize $\mathbf{C}$
  while user inputs a pair of curves γ_{in}, η_{in} **do**
    $\bar{\mathbf{\Phi}}_{\text{in}} \leftarrow \mathbf{P}^M_{\gamma_{\text{in}}}\mathbf{\Phi}$
    $\bar{\mathbf{\Psi}}_{\text{in}} \leftarrow \mathbf{P}^N_{\eta_{\text{in}}}\mathbf{\Psi}$
    $\mathbf{T}_1 \leftarrow \mathbf{T}_1 + 2ns\bar{\mathbf{\Psi}}_{\text{in}}^\top\bar{\mathbf{\Psi}}_{\text{in}}$
    $\mathbf{B} \leftarrow \mathbf{B} + 2\bar{\mathbf{\Psi}}_{\text{in}}^\top\bar{\mathbf{\Phi}}_{\text{in}}$
    $\mathbf{C} \leftarrow$ CONJ GRAD($\mathbf{C}$, $\mathbf{\Lambda}_M$, $\mathbf{\Lambda}_N$, α, $\mathbf{T}_1$, $\mathbf{T}_2$, $\mathbf{B}$)

tively, and we used the fact that $\mathbf{\Lambda}_M, \mathbf{\Lambda}_N$ are diagonal to simplify expressions.

A new curve provided by the user adds the $(q+1)$-st terms $\bar{\mathbf{\Psi}}_{q+1}^\top\bar{\mathbf{\Psi}}_{q+1}\mathbf{C}$ and $\bar{\mathbf{\Psi}}_{q+1}^\top\bar{\mathbf{\Phi}}_{q+1}$ to the summations in $\mathbf{H}(\mathbf{C})$ and $\mathbf{B}$, respectively. Furthermore, the q matrix products $\bar{\mathbf{\Psi}}_\ell^\top\bar{\mathbf{\Psi}}_\ell$ and $\bar{\mathbf{\Psi}}_\ell^\top\bar{\mathbf{\Phi}}_\ell$ of size $k \times k$ can be precomputed as they do not change in a typical workflow in which the user adds curves. With these observations, we can formulate the conjugate gradients algorithm to find the optimal functional map matrix $\mathbf{C}$. Algorithm 1 describes the interactive update procedure discussed above. The conjugate gradients algorithm is elaborated in Algorithm 2.

User feedback. After updating the functional map, the user needs to be able to assess the quality of the resulting map. Unfortunately, conversion into a pointwise map of sufficient quality is not possible at interactive rates using the current available methods (in particular, the method of [ESBC18] that we use in our system). It is possible, however, to visualize the functional map by directly transporting texture mapping coordinates from the source shape M to the target shape N.

Algorithm 2 Conjugate Gradients

function CONJ GRAD($\mathbf{C}$, $\mathbf{\Lambda}_M$, $\mathbf{\Lambda}_N$, α, $\mathbf{T}_1$, $\mathbf{T}_2$, $\mathbf{B}$)
  $\mathbf{R}_1 \leftarrow \mathbf{B}$ - H($\mathbf{C}$, $\mathbf{T}_1$, $\mathbf{T}_2$, $\mathbf{\Lambda}_M$, $\mathbf{\Lambda}_N$, α)
  $\mathbf{P}_1 \leftarrow \mathbf{R}_1$
  $\gamma_1 \leftarrow \sum_{ij}(\mathbf{R}_1 \odot \mathbf{R}_1)_{ij}$
  for $k \leftarrow 1 \dots n_{\text{iter}}$ **do**
    $\alpha_k \leftarrow \dfrac{\gamma_k}{\sum_{ij}(\mathbf{P}_k \odot \text{H}(\mathbf{C}_k, \mathbf{T}_1, \mathbf{T}_2, \mathbf{\Lambda}_M, \mathbf{\Lambda}_N, \alpha))_{ij}}$
    $\mathbf{C}_{k+1} \leftarrow \mathbf{C}_k + \alpha_k \cdot \mathbf{P}_k$
    $\mathbf{R}_{k+1} \leftarrow \mathbf{R}_k - \alpha_k \cdot \text{H}(\mathbf{P}_k, \mathbf{T}_1, \mathbf{T}_2, \mathbf{\Lambda}_M, \mathbf{\Lambda}_N, \alpha)$
    $\gamma_{k+1} \leftarrow \sum_{ij}(\mathbf{R}_{k+1} \odot \mathbf{R}_{k+1})_{ij}$
    if $\gamma_{k+1} \leq 10^{-6}$ **then return** $\mathbf{C}_{k+1}$
    else
      $\beta_k = \frac{\gamma_{k+1}}{\gamma_k}$
      $\mathbf{P}_{k+1} \leftarrow \mathbf{R}_{k+1} + \beta_k \cdot \mathbf{P}_k$
  return $\mathbf{C}_n$
function H($\mathbf{C}$, $\mathbf{T}_1$, $\mathbf{T}_2$, $\mathbf{\Lambda}_M$, $\mathbf{\Lambda}_N$, α)
  return $\mathbf{T}_1\mathbf{C} + \mathbf{C}\mathbf{T}_2 - \alpha(\mathbf{\Lambda}_M\mathbf{C}\mathbf{\Lambda}_N + \mathbf{\Lambda}_N\mathbf{C}\mathbf{\Lambda}_M)$

We assume that the source shape M has associated texture mapping coordinates represented as a vector-valued function $(u,v) : M \to [0,1]^2$ on the manifold. In the discrete setting, texture mapping coordinates are represented as an $m \times 2$ matrix $\mathbf{U}$. Application of the functional map is performed by first computing the representation in the Laplace basis of the texture mapping coordinates on M, applying the matrix $\mathbf{C}$ to the obtained coefficients, and then reconstructing them on the manifold N:

$$\mathbf{TU} = \mathbf{\Psi}\mathbf{C}\mathbf{\Phi}^\top\mathbf{A}_M\mathbf{U}.$$

This operation is cheap and can be performed at interactive rates.

Visualizing the transported texture on the target mesh directly gives the user feedback about where the map needs to be improved. For example, in Figure 3 the user has added several curves and the texture that is transported by the resulting functional map, adequately maps semantically corresponding parts onto each other. If the user is satisfied with the quality of the map, the interface has

Figure 4: *Qualitative comparison of our method (rightmost column) to [NO17] and [ESBC18] on non-isometric shapes, visualized with texture mapping. The reference shape is shown in the leftmost image. The magenta curves/dots indicate the user-input given to the respective method The correspondences recovered with our method are more accurate as it is able to deal with significant metric distortions.*

an option allowing to run the slower higher-quality point-to-point conversion method of [ESBC18].

5. Applications and Results

In this section, we show how curve-constrained functional maps can be used to obtain high quality point-to-point correspondences (Section 5.1). Furthermore, we provide qualitative and quantitative evaluation of our method and compare to previous approaches (Section 5.2). In all the qualitative correspondence results shown, unless indicated otherwise, the final pointwise correspondences are obtained from functional maps with the method of [ESBC18]. For our results we set $k = 120$ (number of eigenfunctions) and $\alpha = 0.5$.

5.1. Point-to-point correspondences

As with previous work on functional maps, our functional maps can be used as input to pointwise correspondence computation. Beyond the simple map recovery technique proposed in the original functional maps paper [OBCS*12], Ezuz et al. [ESBC18] propose a method that, given pointwise correspondences from M to N and vice versa, refines them by minimizing a harmonic-style energy. We use their method as the final stage of our pipeline as follows. First, we compute the functional maps $\mathbf{C}_{MN}$ and $\mathbf{C}_{NM}$ in both directions between the shapes. Second, we extract an initial pointwise correspondence by nearest neighbor search as in [OBCS*12]. Finally, we refine the initial correspondences with the method of [ESBC18]. Results are shown in Figures 1, 4, 5, and 6.

In Figure 4, we provide a qualitative comparison to [ESBC18] and [NO17]. These are automatic approaches which are based on surface descriptors and require only sparse user input in form of point correspondences. We use the authors implementations to compute the results. For the comparison to [ESBC18] (Figure 4), we use their entire pipeline, employing 200-dimensional wave kernel signatures (wks) and wave-kernel map (wkm) functions (200 dimensions each) constructed on 21 landmark points picked by hand, leading to 4400 function-preservation constraints. For the method of [NO17] (Figure 4), we used the authors' implementation based on 200 wave-kernel signature functions and 6 wave-kernel-map functions (with 200 dimensions each, also provided by hand), i.e. with 1400 function preservation constraints, to compute the functional maps in both directions, and then proceed as in our method (first compute the initial correspondence as in [OBCS*12] and then refine with [ESBC18]). The functional map computed with our method gets as input 18 feature curve constraints shown in Figure 4. Methods for recovering pointwise correspondences from functional maps like [ESBC18] are usually computationally intensive. Since our method allows the user to control the quality of the functional map at runtime, the initialization of the pointwise correspondence recovery methods is reliable, implying that the user would typically have to run it only once at the end of the interactive session. Even though the corresponding parts are very non-isometric (see e.g. the legs of the bear and the human in Figure 4), the correspondence obtained with our curve-based approach is accurate. In contrast, automatic methods with little user input (usually in the form of point correspondences for the wave-kernel map) output inaccurate correspondences when shapes vary strongly in terms of conformal factor and other distortion measures.

Figure 5: *Correspondences between surfaces with* extremely distorting *constraints and varied geometry, visualized with texture mapping. Feature curves used as the input to our optimization procedure are shown in red. Note in the left example that we map the head of the wolf onto the tail of the cat and vice versa.*

Figure 5 shows maps between very challenging mesh pairs, including extremely non-isometric deformations and high-frequency details. While we do not expect to find low-distortion maps in these cases, our correspondences obey the curve constraints and are relatively meaningful.

5.2. Correspondence Accuracy

To evaluate the quality of the correspondence, we first recover the pointwise correspondence from the functional maps using nearest neighbors [OBCS*12], and then plot the percentage of correspondences found within a certain geodesic radius (in % of the target shape geodesic diameter) using the Princeton benchmark [KLF11]. The results are depicted in Figure 10 and 11, where we show the average correspondence accuracy on the high-resolution *TOSCA* [BBK08] and *FAUST* [BRLB14] datasets, as well as a far-from-

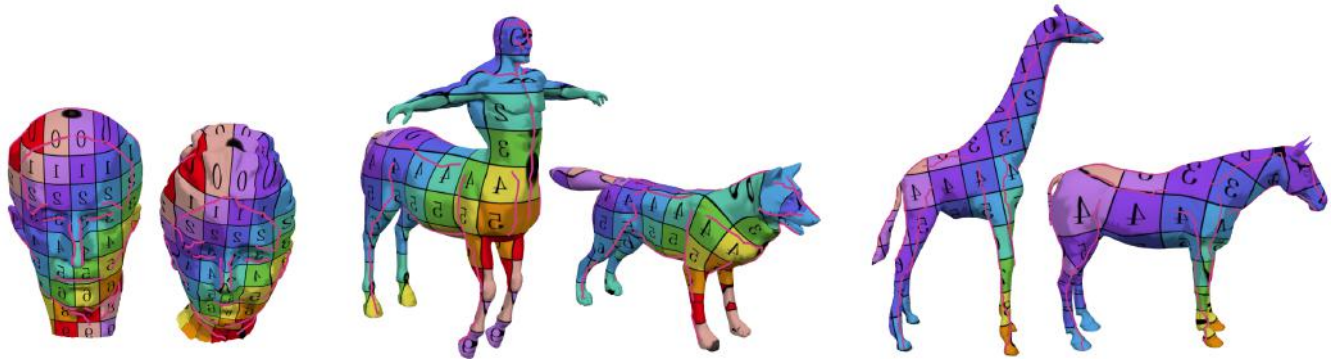


Figure 6: *Examples of correspondence results produced by our method, visualized with texture mapping. Feature curves used as the input to our optimization procedure are shown in red.*

isometric dataset containing a pair of horse and elephant shapes (cf. Figure 7) (Figure 12). These datasets contain pre-labeled point-correspondences with which we are able to evaluate the quality of our computed maps. In Figure 7, we show the labeled feature curve correspondences. The texture is transported from the source shape (indicated by a box) to the target shapes with the functional map computed based on the feature curve constraints.

We compare our results to [NO17] and to an interactive alternative using just the descriptor-based linear solver in [OBCS*12]. [NO17] is the recent state-of-the-art automatic functional map computation method. While it achieves high correspondence accuracy, this approach is computationally heavy and cannot run at interactive speeds. A possible interactive alternative to our approach can be implemented by solving the linear system $\mathbf{C}\hat{\mathbf{F}} = \hat{\mathbf{G}}$, i.e. the descriptor preservation constraints presented in [OBCS*12]. For the functional map computation we provide the methods with 1400 wks/wkm-function preservation constraints as described above.

We vary the following test parameters in evaluating correspondence accuracy:

Increasing number of feature curves. We plot the correspondence accuracy for functional maps based on an increasing number of curves. These pre-labeled corresponding curves follow features on the mesh (cf. Figure 7). For a given number of curves we run the correspondence test multiple times by selecting a random subset from the prelabled feature curves. Examples for all datasets are given in Figure 7. Increasing the number of curves used as input, the correspondence accuracy is improved. We also observe that functional maps based on only one pair of corresponding curves already produces more accurate correspondences than that based on descriptor preservation solving $\mathbf{C}\hat{\mathbf{F}} = \hat{\mathbf{G}}$.

By increasing the number of curves we can even achieve a higher match rate than [NO17], even though the shapes in the TOSCA and FAUST datasets are nearly isometric deformations of one another. For these nearly-isometric pairs, state-of-the-art functional map methods are expected to perform well.

Imperfect correspondence information. The solid curves in Figure 11 use perfectly corresponding curves (i.e. they follow the same path of point-correspondences). In contrast, the dashed plots are found by selecting the same start- and end-point and tracing the intermediate path with shortest path search. We observe that there is no significant difference in the results: The shortest path search provides a reliable method to select corresponding curves.

To evaluate the effect of imperfect correspondence information we plot the geodesic error for correspondences found with [OBCS*12] based on functional maps with imperfect input curve correspondences. In Figure 9 the 3 blue curves are perfectly corresponding while the magenta curves on the target are found by shortest path search on the target mesh between corresponding start- and end-points. Since start- and end-points lie far apart the correspondences are inexact. We show the geodesic error in a range from 0 (white) to 0.5 (red) which is obtained from the functional maps with exact (top row) and inexact (bottom row) feature curve correspondences for an increasing number of curves (left to right). In regions where the curves deviate a lot the geodesic error is higher (e.g. the belly of the cat). However, although locally the error is higher, the average geodesic distance is still quite low (0.029 with inexact correspondence information and 0.0043 with exact correspondences for 3 curves).

Far from isometric and baseline test. We plot the correspondence quality for the horse-elephant dataset in Figure 10(c), where we show the match characteristics up to a total geodesic distance of 0.5. With the curve-constrained functional maps we can achieve much higher match accuracy compared to previous methods. This dataset is especially challenging due to large distortion (see Figure 8 for a visualization of the ground-truth map).

In Figure 12 we show the match characteristics on the SHREC [VtH07] dataset. We use all meshes from the SHREC dataset where a user can unambiguously label 10 corresponding curves without the need to see the ground truth (e.g. we did not use rotationally symmetric vases). As above we compare to [NO17] with 1400 function preservation constraints (wkm and wks). Secondly, we perform a baseline comparison by providing the 10 feature curves as functions to [NO17]. The functions are set to -1 everywhere, but at the vertices along the feature curve. To each vertex along the curve we assign the value of the arc-length parametrization (normalized to a total length of 1). Correspondences computed with

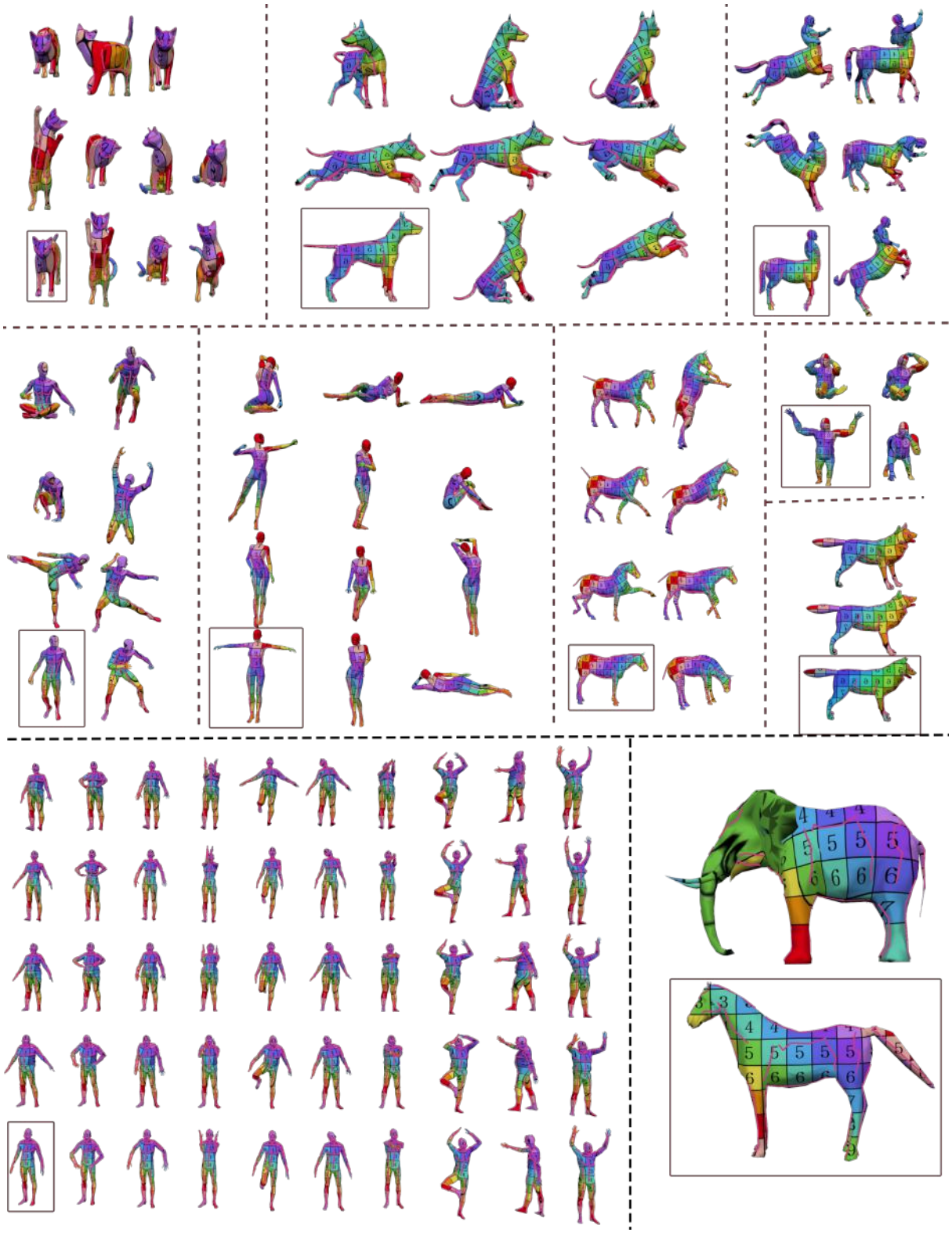


Figure 7: *TOSCA, FAUST, and horse-elephant dataset labeled with corresponding feature curves. Since these datasets have prelabled point-correspondences, matching feature curves can be defined by following the same vertices's along the curve. The texture coordinates are mapped by the functional map that is computed based on the feature curve constraints. The source shape for each shapes group is indicated by a box. We only show half of the FAUST shapes here.*

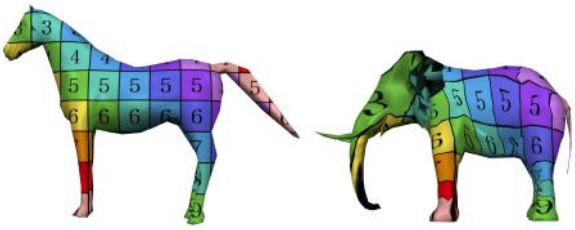
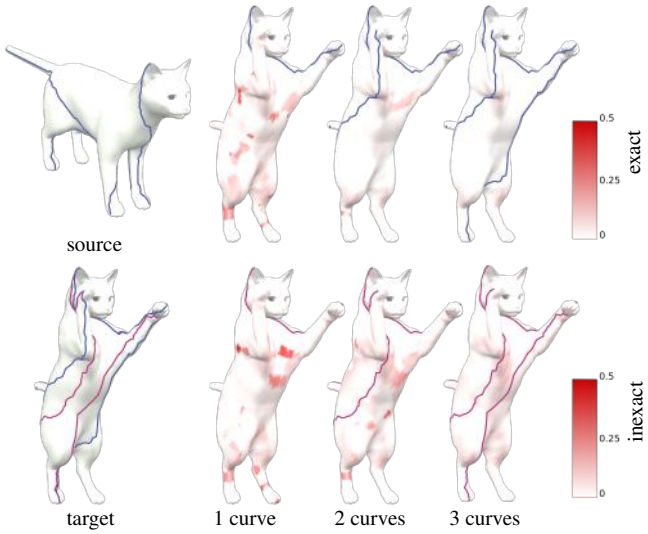


Figure 8: *Ground-truth correspondences of the far from isometric elephant-horse dataset.*



Curves	1 curve	2 curves	3 curves
exact	0.0387	0.0243	0.0043
inexact	0.0432	0.0368	0.0298

Figure 9: *Geodesic error for imperfect correspondence information with increasing number of curves. We select 3 curves with distant start and end point. The blue curves indicate perfect correspondence, while the magenta curves show the shortest path curves found on the target mesh between start- and end-point. We visualize the geodesic error for an increasing number of corresponding curves in a range from 0 (white) to 0.5 (red). The geodesic error is higher at points where the curves deviate a lot (e.g. at the belly of the cat). The average geodesic error is given in the table above.*

functional maps based on both the baseline test and the wave-kernel descriptors show higher geodesic distances to the ground truth than our method.

5.3. Timing

In Figure 13, we compare our timings for optimization and preprocessing to the results in [NO17] for a different number of curve constraints. In this comparison we provide the same user input to our method and to [NO17]. Results are computed on a commodity laptop on meshes with 27,560 (pig) and 33,638 (dog) vertices. For [NO17], we use 200 functions of the wave-kernel signature and the start and endpoints of the feature curves (user input) for the computation of the wave-kernel map with 200 functions each; for [NO17], the selection of these points at interactive speeds is not possible. The results with our method are obtained in under a second. Furthermore, we can observe that with our method we get slightly improved results (see e.g. the front foot and tail for 10 curves and the mouth and tail for 20 curves).

We compare the runtime of our conjugate gradients implementation with the quasi-Newton solver used by [NO17]. For this we optimize our curve based functional map objective (5) with both methods for a different number of curves. In contrast to [NO17] our conjugate gradients implementation provides feedback at interactive rates.

Number of Curves	Quasi-Newton Solver	Conjugate Gradients
1 curve	3.955s	0.145s
10 curves	4.724s	0.624s
20 curves	5.077s	0.642s

Solving a dense linear system of the form $\mathbf{Hx} = \mathbf{b}$ with $\mathbf{H} \in \mathbb{R}^k$ has a complexity of $\mathcal{O}(k^6)$. Our iterative implementation converges quickly leading to interactive response times.

We show the runtime of the initialization and optimization procedure in Figure 14 for increasing k. The images (top) show the texture coordinates transported with the respective functional map. Although the map is slightly sharpened for increasing k, for $k > 20$ no significant changes are visible in the qualitative comparison. For $k = 200$ the runtime increases beyond interactive rates. However, for the visualization of the functional map less eigenfunctions are sufficient.

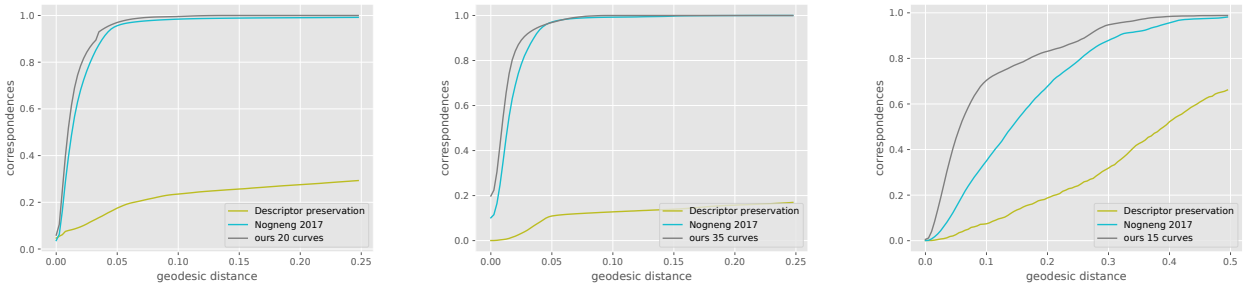
6. Conclusion

In this paper, we presented an interactive approach to constructing functional maps. While related work opts to find increasingly elaborate and computationally intensive approaches to functional maps, we present a method which is both simple (easy to implement) and efficient such that it runs at interactive speeds. In our tests we find that by adding feature curve constraints we can outperform state-of-the art automatic approaches in terms of correspondence accuracy.

This new interactive direction for functional maps has two major benefits compared to previous work. First, it allows to compute smooth maps between semantically similar objects, even if they vary geometrically. Secondly, results are obtained at interactive speeds and can be edited and evaluated by a user. For non-expert users, feature curve based functional maps are especially useful, since they avoid the requirement to engineer descriptors specific to the regarded shape family.

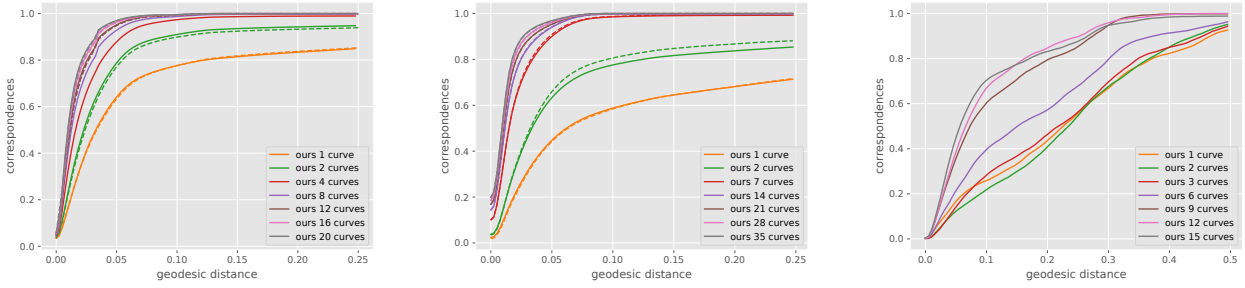
Limitations and future work. One issue we observed is that the feedback of the interactive tool given by the texture coordinates mapped by the functional map can be inaccurate along (non-smooth) seams in the texture. Nonetheless, it provides an effective way to roughly evaluate the quality of the map during its design. Then in a followup step, the user can compute an exact pointwise map with the provided functional map.

Furthermore, our results show that several curve constraints are required to obtain a meaningful functional map (i.e. one curve-pair is usually not sufficient). However, the input the user has to



TOSCA FAUST Non-isometric deformations

Figure 10: *Correspondence quality of different methods on TOSCA (left), FAUST (center), and the Horse-Elephant pair (right). Shown are the results of our method, descriptor preservation constraints presented in [OBCS*12], and [NO17]. Point correspondences were obtained from the functional maps using nearest neighbors as in [OBCS*12]. The advantage of our method is especially pronounced on non-isometric shapes.*



TOSCA FAUST Horse-Elephant

Figure 11: *Correspondence quality of our method on TOSCA (left), FAUST (center), and the Horse-Elephant pair (right). Point correspondences were obtained from the functional maps using nearest neighbors as in [OBCS*12]. We used perfectly corresponding curves (solid) and shortest path curves computed between perfectly corresponding endpoints (dashed). Almost no degradation in performance is observed in the latter case, allowing us to conclude that our shortest path method approximates the feature curves sufficiently well.*

provide is still quite sparse. We have shown that inexact curve constraints (based on automatically traced shortest paths between user-provided endpoints) are sufficient to obtain high-quality maps. Hence, curve tracing can be performed automatically given the start- and end-points of the curves.

Since the computation of the maps is very efficient and the resulting correspondence accuracy is comparable to state-of the art methods, an interesting direction of research would be to automatically detect semantically corresponding feature curves. Inspired by previous work on functional maps, it would also be interesting to investigate coupled functional maps where multiple objects are labeled with corresponding curves.

We also observe that is hard to provide meaningful curve-based constraints if one shape has parts that are absent in another one (e.g. elephant trunks) or very different from the corresponding parts (e.g., elephant ears). In such cases it is hard to obtain a reasonable bijective map. Previous work has dealt with the problem of partial functional maps [RCB*17]. An interesting direction for future research might be to combine these approaches with our method.

Acknowledgements

The research leading to these results has received funding from the German Academic Exchange Service with the scholarship *FIT weltweit* and the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement n° [340884]. M. Bronstein is supported in part by the ERC Consolidator Grant No. 724228 (LEMAN), Google Faculty Research Awards, an Amazon AWS Machine Learning Research grant, an Nvidia equipment grant, a Radcliffe Fellowship at the Institute for Advanced Study, Harvard University, and a Rudolf Diesel Industrial Fellowship at the Institute for Advanced Study, TU Munich. J. Solomon acknowledges the generous support of Army Research Office grant W911NF-12-R0011 ("Smooth Modeling of Flows on Graphs"), from the MIT Research Support Committee, from the MIT–IBM Watson AI Lab, from the Skoltech–MIT Next Generation Program, and from an Amazon Research Award. We would like to thank Danielle Ezuz and Emanuele Rodola for providing code and meshes, which we used for our results. Furthermore we thank Jan Möbius for creating and maintaining the

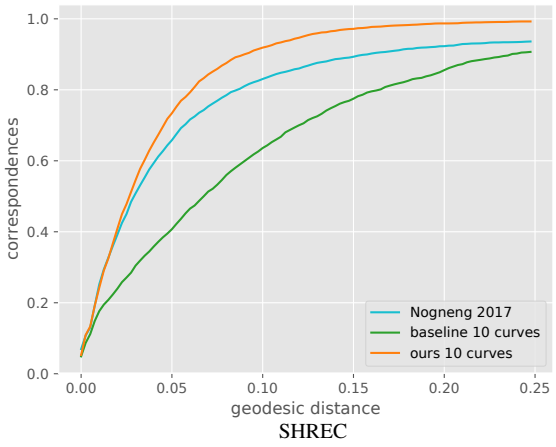
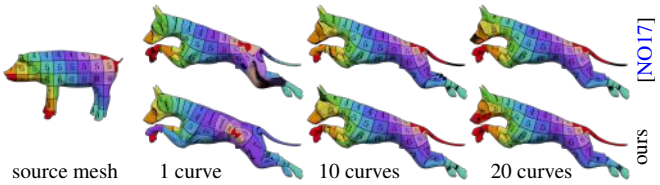


Figure 12: *Correspondence quality of our method and [NO17] with descriptors (wks and wkm) and curve functions (baseline test) on the SHREC dataset [VtH07]. Point correspondences were obtained from the functional maps using nearest neighbors as in [OBCS*12]. Our method shows a higher correspondence accuracy than the compared approaches.*



Method	Number of Curves	Preprocessing	Optimization
[NO17]	1 curve / 2 points	34.823s	24.223s
[NO17]	10 curve / 20 points	267.815s	99.592s
[NO17]	20 curve / 40 points	555.921s	147.252s
our method	1 curve	0.003s	0.145s
our method	10 curves	0.008s	0.624s
our method	20 curves	0.007s	0.642s

Figure 13: *Timings and qualitative comparison to [NO17]. We provide a point-to-point mapping as input to the point-to-point reconstruction method [ESBC18] for our method and [NO17]. We use the same correspondences (start- and endpoints of the curves) for the wave-kernel-map compuatation in [NO17]. The table below gives the timings for preprocessing (setting up descriptors and matrices) and the functional map optimization. Note that the selection of point-correspondences cannot be performed at interactive speeds for [NO17]. Furthermore, parts of the mesh show a higher correspondence accuracy with our method (see e.g. the front foot and tail for 10 curves and the mouth and tail for 20 curves).*

geometry processing framework OpenFlipper [MK12] as well as the reviewers for their insightful comments.

References

[ASC11] AUBRY M., SCHLICKEWEI U., CREMERS D.: The wave kernel signature: A quantum mechanical approach to shape analysis. In *Proc. ICCV Workshops* (2011), pp. 1626–1633. 2

[B*17] BOYARSKI A., ET AL.: Efficient deformable shape correspondence via kernel matching. In *Proc. 3DV* (2017). 3

[BBK08] BRONSTEIN A., BRONSTEIN M., KIMMEL R.: *Numerical Geometry of Non-Rigid Shapes*, 1 ed. Springer Publishing Company, Incorporated, 2008. 6

[BK10] BRONSTEIN M. M., KOKKINOS I.: Scale-invariant heat kernel signatures for non-rigid shape recognition. In *Proc. CVPR* (2010), pp. 1704–1711. 2

[BMR*16] BOSCAINI D., MASCI J., RODOLÀ E., BRONSTEIN M. M., CREMERS D.: Anisotropic diffusion descriptors. *Computer Graphics Forum 35*, 2 (2016), 431–441. 2

[BMRB16] BOSCAINI D., MASCI J., RODOLÀ E., BRONSTEIN M.: Learning shape correspondence with anisotropic convolutional neural networks. In *Proc. NIPS* (2016). 2

[BRLB14] BOGO F., ROMERO J., LOPER M., BLACK M. J.: FAUST: Dataset and evaluation for 3D mesh registration. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (Piscataway, NJ, USA, June 2014), IEEE. 6

[CHK13] CAMPEN M., HEISTERMANN M., KOBBELT L.: Practical anisotropic geodesy. *Computer Graphics Forum 32*, 5 (2013), 63–71. 4

[DVVR07] DEMARSIN K., VANDERSTRAETEN D., VOLODINE T., ROOSE D.: Detection of closed sharp edges in point clouds using normal estimation and graph theory. *Computer-Aided Design 39*, 4 (2007), 276–283. 2

[EBC17] EZUZ D., BEN-CHEN M.: Deblurring and denoising of maps between shapes. *Computer Graphics Forum 36*, 5 (2017), 165–174. 3

[EKB*15] EYNARD D., KOVNATSKY A., BRONSTEIN M. M., GLASHOFF K., BRONSTEIN A. M.: Multimodal manifold analysis by simultaneous diagonalization of laplacians. *Trans. PAMI 37*, 12 (2015), 2505–2517. 3

[ERGB16] EYNARD D., RODOLA E., GLASHOFF K., BRONSTEIN M. M.: Coupled functional maps. In *Proc. 3DV* (2016). 3

[ESBC18] EZUZ D., SOLOMON J., BEN-CHEN M.: Reversible harmonic maps between discrete surfaces. *arXiv:1801.02453* (2018). 1, 3, 5, 6, 11

[GLK18] GEHRE A., LIM I., KOBBELT L.: Feature curve co-completion in noisy data. *Computer Graphics Forum 37*, 2 (2018). 2

[GSTOG16] GANAPATHI-SUBRAMANIAN V., THIBERT B., OVSJANIKOV M., GUIBAS L.: Stable region correspondences between non-isometric shapes. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 121–133. 2

[HPW05] HILDEBRANDT K., POLTHIER K., WARDETZKY M.: Smooth feature lines on surface meshes. In *Proc. SGP* (2005). 2

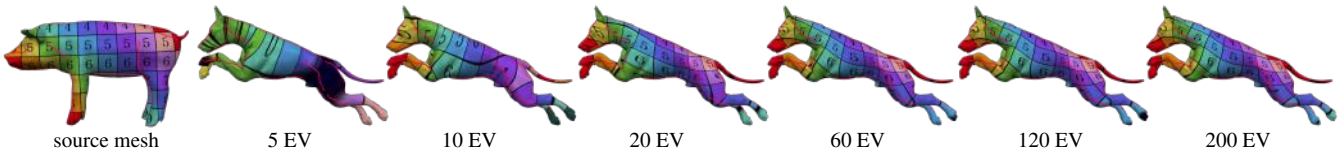
[HS52] HESTENES M. R., STIEFEL E.: *Methods of conjugate gradients for solving linear systems*, vol. 49. NBS Washington, DC, 1952. 4

[HWG14] HUANG Q., WANG F., GUIBAS L.: Functional map networks for analyzing and exploring large shape collections. *TOG 33*, 4 (2014), 36. 3

[KBB*13] KOVNATSKY A., BRONSTEIN M. M., BRONSTEIN A. M., GLASHOFF K., KIMMEL R.: Coupled quasi-harmonic bases. 439–448. 3

[KBLB12] KOKKINOS I., BRONSTEIN M. M., LITMAN R., BRONSTEIN A. M.: Intrinsic shape context descriptors for deformable shapes. In *Proc. CVPR* (2012). 2

[KLF11] KIM V., LIPMAN Y., FUNKHOUSER T.: Blended intrinsic maps. *ACM Transactions on Graphics (Proc. SIGGRAPH) 30*, 4 (July 2011). 6



Number of EV (k)	5	10	20	60	120	200
Eigenvector computation	0.526s	0.610s	0.789s	1.618s	5.148s	8.993s
Preprocessing	0.005s	0.005s	0.005s	0.006s	0.007s	0.012s
Optimization	0.005s	0.006s	0.008s	0.059s	0.642s	2.788s

Figure 14: *Timings and qualitative comparison for different numbers of Laplace eigenvectors (EV). The qualitative comparison shows the texture coordinates that are transported by the functional map computed with the respective number of eigenfunctions and 20 curve constraints. Although maps are slightly sharper with higher amount of eigenfunctions, the qualitative difference between results is not significant for $k > 20$.*

[LB14] LITMAN R., BRONSTEIN A. M.: Learning spectral descriptors for deformable shape correspondence. *IEEE Trans. PAMI 36*, 1 (2014), 171–180. 2

[LGB*13] LIAN Z., GODIL A., BUSTOS B., DAOUDI M., HERMANS J., KAWAMURA S., KURITA Y., LAVOUÉ G., VAN NGUYEN H., OHBUCHI R., ET AL.: A comparison of methods for non-rigid 3d shape retrieval. *Pattern Recognition 46*, 1 (2013), 449–461. 2

[LRR*17] LITANY O., REMEZ T., RODOLA E., BRONSTEIN A. M., BRONSTEIN M. M.: Deep functional maps: Structured prediction for dense shape correspondence. In *Proc. ICCV* (2017), vol. 2, p. 8. 2

[LWWS15] LI C., WAND M., WU X., SEIDEL H.-P.: Approximate 3d partial symmetry detection using co-occurrence analysis. In *Proc. 3DV* (2015). 2

[LZH*07] LAI Y.-K., ZHOU Q.-Y., HU S.-M., WALLNER J., POTTMANN H.: Robust feature classification and editing. *IEEE Trans. Visualization and Computer Graphics 13*, 1 (2007). 2

[MBBV15] MASCI J., BOSCAINI D., BRONSTEIN M., VANDERGHEYNST P.: Geodesic convolutional neural networks on riemannian manifolds. In *Proc. ICCV Workshops* (2015). 2

[MK12] MÃŰBIUS J., KOBBELT L.: Openflipper: An open source geometry processing and rendering framework. In *Curves and Surfaces*, Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2012. 11

[NMR*18] NOGNENG D., MELZI S., RODOLÀ E., CASTELLANI U., BRONSTEIN M., OVSJANIKOV M.: Improved functional mappings via product preservation. *Computer Graphics Forum 37* (2018). 3

[NO17] NOGNENG D., OVSJANIKOV M.: Informative descriptor preservation via commutativity for shape matching. *Computer Graphics Forum 36*, 2 (2017), 259–267. 3, 6, 7, 9, 10, 11

[OBCS*12] OVSJANIKOV M., BEN-CHEN M., SOLOMON J., BUTSCHER A., GUIBAS L.: Functional maps: A flexible representation of maps between shapes. *TOG 31*, 4 (July 2012), 1–11. 2, 3, 6, 7, 10, 11

[OMMG10] OVSJANIKOV M., MÉRIGOT Q., MÉMOLI F., GUIBAS L.: One point isometric matching with the heat kernel. *Computer Graphics Forum 29*, 5 (2010), 1555–1564. 2

[PBB*13] POKRASS J., BRONSTEIN A. M., BRONSTEIN M. M., SPRECHMANN P., SAPIRO G.: Sparse modeling of intrinsic correspondences. *Computer Graphics Forum 32*, 2pt4 (2013), 459–468. 3

[RCB*17] RODOLÀ E., COSMO L., BRONSTEIN M. M., TORSELLO A., CREMERS D.: Partial functional correspondence. *Computer Graphics Forum 36*, 1 (2017), 222–236. 3, 10

[Rus07] RUSTAMOV R. M.: Laplace–B|eltrami eigenfunctions for deformation invariant shape representation. In *Proc. SGP* (2007), pp. 225–233. 2

[SOG09] SUN J., OVSJANIKOV M., GUIBAS L.: A concise and provably informative multi-scale signature based on heat diffusion. *Computer graphics forum 28*, 5 (2009), 1383–1392. 2

[Sol15] SOLOMON J.: *Numerical Algorithms: Methods for Computer Vision, Machine Learning, and Graphics*. CRC Press, 2015. 4

[STDS14] SALTI S., TOMBARI F., DI STEFANO L.: SHOT: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding 125* (2014), 251–264. 2

[VKZHCO11] VAN KAICK O., ZHANG H., HAMARNEH G., COHEN-OR D.: A survey on shape correspondence. *Computer Graphics Forum 30*, 6 (2011), 1681–1707. 2

[VLR*17] VESTNER M., LITMAN R., RODOLÀ E., BRONSTEIN A., CREMERS D.: Product manifold filter: Non-rigid shape correspondence via kernel density estimation in the product space. In *Proc. CVPR* (2017). 3

[VtH07] VELTKAMP R., TER HAAR F.: Shrec 2007 3d shape retrieval contest dept of info and comp. sci. utrecht univ. 7, 11

[WB01] WATANABE K., BELYAEV A. G.: Detection of salient curvature features on polygonal surfaces. *Computer Graphics Forum 20*, 3 (2001), 385–392. 2

[WG09] WEINKAUF T., GÜNTHER D.: Separatrix persistence: Extraction of salient edges on surfaces using topological methods. *Computer Graphics Forum 28*, 5 (2009), 1519–1528. 2

[YBS05] YOSHIZAWA S., BELYAEV A., SEIDEL H.-P.: Fast and robust detection of crest lines on meshes. In *Proc. ACM Symp. Solid and Physical Modeling* (2005), pp. 227–232. 2