Understanding and Improving Features Learned in Deep Functional Maps

Souhaib Attaiki
LIX, École Polytechnique, IP Paris
attaiki@lix.polytechnique.fr

Maks Ovsjanikov
LIX, École Polytechnique, IP Paris
maks@lix.polytechnique.fr

Abstract

Deep functional maps have recently emerged as a successful paradigm for non-rigid 3D shape correspondence tasks. An essential step in this pipeline consists in learning feature functions that are used as constraints to solve for a functional map inside the network. However, the precise nature of the information learned and stored in these functions is not yet well understood. Specifically, a major question is whether these features can be used for any other objective, apart from their purely algebraic role in solving for functional map matrices. In this paper, we show that under some mild conditions, the features learned within deep functional map approaches can be used as point-wise descriptors and thus are directly comparable across different shapes, even without the necessity of solving for a functional map at test time. Furthermore, informed by our analysis, we propose effective modifications to the standard deep functional map pipeline, which promote structural properties of learned features, significantly improving the matching results. Finally, we demonstrate that previously unsuccessful attempts at using extrinsic architectures for deep functional map feature extraction can be remedied via simple architectural changes, which encourage the theoretical properties suggested by our analysis. We thus bridge the gap between intrinsic and extrinsic surface-based learning, suggesting the necessary and sufficient conditions for successful shape matching. Our code is available at https://github.com/pvnieo/clover*.*

1. Introduction

Computing dense correspondences between 3D shapes is a classical problem in Geometry Processing, Computer Vision, and related fields, and remains at the core of many tasks including statistical shape analysis [9, 56], registration [86], deformation [8] or texture transfer [15] among others.

Since its introduction, the functional map (fmap) pipeline [53] has become a de facto tool for addressing this problem. This framework relies on representing correspondences as linear transformations across functional spaces, by encoding them as small matrices using the Laplace-Beltrami
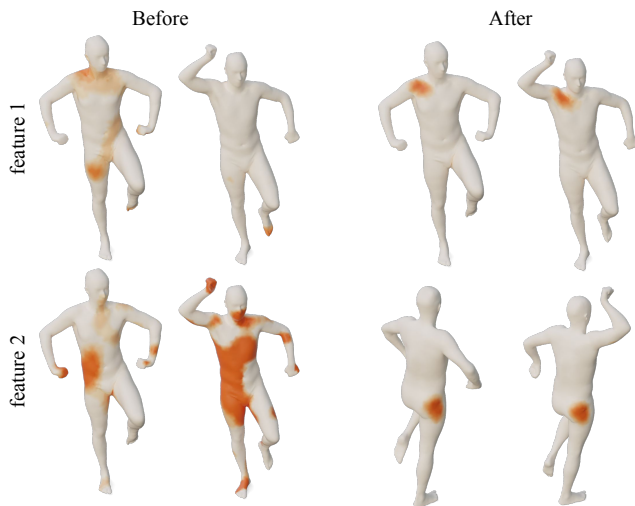

Figure 1. *Left:* Features learned in existing deep functional map pipelines are used in a purely *algebraic manner* as constraints for a linear system, thus lacking interpretability and clear geometric content. *Right:* We propose a theoretically-justified modification to this pipeline, which leads to learning *robust* and *repeatable* features that enable matching via nearest neighbor search.

basis. Methods based on this approach have been successfully applied with hand-crafted features [6, 71, 76] to many scenarios including near-isometric, [21, 32, 48, 65, 75], non-isometric [18, 36] and partial [13, 39, 40, 67, 82, 83] shape matching. In recent years, a growing body of literature has advocated improving the functional map pipeline by using *deeply* learned features, pioneered by [38], and built upon by many follow-up works [4, 17, 20, 31, 43, 44, 72, 73]. In all of these methods, the learned features are only used to constrain the linear system when estimating the functional maps inside the network. Thus, no attention is paid to their geometric nature, or potential utility beyond this purely algebraic role.

On the other hand, features learned in other deep matching paradigms are the main focus of optimization, and they represent either robust descriptive geometric features that are used directly for matching using nearest neighbor search [7, 14, 26, 37, 41, 84, 85], or as distributions that, at every point, are interpreted as vertex indices on some tem-

plate shape [46, 50, 57], or a deformation field that is used to deform the input shape to match a template [28].

In contrast, feature (also known as "probe" [54]) functions, within deep functional maps are used purely as an optimisation tool and thus the information learned and stored in these functions is not yet well-understood. In this work, we aim to show that features in deep functional maps networks can, indeed, have geometric significance and that, under certain conditions, they are directly comparable across shapes, and can be used for matching simply via nearest neighbor search, see Fig. 1.

Specifically, we introduce the notion of feature completeness and show that under certain mild conditions, extracting a pointwise map from a functional map or via the nearest neighbor search between learned features leads to the same result. Secondly, we propose a modification of the deep functional map pipeline, by imposing the learned functional maps to satisfy the conditions imposed by our analysis. We show that this leads to a significant improvement in accuracy, allowing state-of-the-art results by simply performing a nearest-neighbor search between features at test time. Finally, based on our theoretical results, we also propose a modification of some extrinsic feature extractors [79, 81], which previously failed in the context of deep functional maps, which improve their overall performance by a significant margin. Since our theoretical results hold for the functional map paradigm in general, they can be incorporated in *any* deep fmap method, hence improving previous methods, and making future methods more robust.

Overall, our contributions can be summarized as follows:

- We introduce the notions of feature completeness and basis aligning functional maps and use them to establish a theoretical result about the nature of features learned in the deep functional map framework.

- Informed by our analysis, we propose simple modifications to this framework, which lead to state-of-the-art results in challenging scenarios.

- Based on our theoretical results, we propose a simple modification to some extrinsic feature extractors, that were previously unsuccessful for deep functional maps, improving their overall accuracy, and bridging the gap between intrinsic and extrinsic surface-based learning.

2. Related Work

Non-rigid shape matching is a very rich and well-established research area, and a full overview is beyond the scope of this paper. Below, we review works that are close to our method and refer the interested reader to recent surveys [10, 12, 29, 30] for a more in-depth treatment.

Functional Maps Since its introduction [52], the functional map pipeline has become a widely-used tool for non-rigid shape matching, being adopted and extended in many follow-up works [21, 21, 24, 48, 51, 59, 65, 68, 72]. The advantage of this approach is that it reduces the optimization of pointwise maps, which are quadratic in the number of vertices, to optimizing small matrices, thus greatly simplifying computational problems. We refer to [54] for an overview.

The original functional map pipeline relied on input feature (probe) functions, which were given a priori [6, 71, 76]. Subsequent research has improved the method by adapting it to partial shapes [13, 39, 68] or using robust regularizers [11, 36, 51, 65] and proposed efficient refinement techniques [48, 55, 62]. In all of these works, fmaps were computed using hand-crafted probe functions, and any information loss in these descriptors hinders downstream optimization.

More recent works have proposed to solve this problem by learning descriptors (probe functions) directly from the data, using deep neural networks. This line of research was initiated by FMNet [38], and extended in many subsequent works [2–4, 17, 19, 20, 31, 37, 69, 72, 73]. These methods learn the probe features directly from the raw geometry, using a neural network, and supervise the learning with a loss on the functional map in the reduced basis.

A parallel line of work has focused on making the learning *unsupervised*, which can be convenient in the absence of ground truth correspondences. Approaches in [31] and [25] have proposed penalizing either the geodesic distortion of the pointwise map predicted by the network or using the cycle consistency loss. Another line of work [16, 69, 72, 73] considered unsupervised training by imposing structural properties on the functional maps in the reduced basis, such as bijectivity, orthonormality, and commutativity of the Laplacian. The authors of [72] have shown that feature learning can be done starting from raw 3D geometry in the weakly supervised setting, where shapes are only approximately rigidly pre-aligned.

In all of these works, features extracted by neural networks have been used to formulate the optimization problem, from which a functional map is computed. Thus, so far no attention has been paid to the geometric nature or other utility of learned probe functions. In contrast, we aim to analyze the conditions under which probe functions can be used for direct pointwise map computation and use this analysis to design improved map estimation pipelines.

We also note briefly a very recent work [37] which has advocated for imposing feature smoothness when learning for non-rigid shape correspondence. However, that work does not use the in-network functional map estimation and moreover lacks any theoretical analysis or justification for its design choices.

Recent Advances in Axiomatic Functional Maps Our results are also related to recent axiomatic methods for functional map-based methods, which *couple* optimization for

the functional map (fmap) with the associated point map (p2p map) [48, 55, 66, 83]. These techniques typically propose to refine functional maps by *iterating* the conversion from functional to pointwise maps and vice versa. This approach was recently summarized in [63], where the authors introduce the notion of functional map "properness" and describe a range of energies that can be optimized via this iterative conversion scheme.

The common denominator between all these methods is the inclusion of pointwise maps in the process of functional map optimization. In this work, we propose a method and a loss that similarly incorporate pointwise map computation. However, we do so in a learning context and show that this leads to significant improvements in the overall accuracy of the deep functional map pipeline.

Learning on Surfaces Multiple methods for deep surface learning have been proposed to address the limitations of handcrafted features in downstream tasks. One type of method is point-based (extrinsic) methods, such as Point-Net [60] and follow-up works [5, 42, 61, 77, 79, 81]. These methods are simple, effective, and widely applicable. However, they often fail to generalize to new datasets or significant pose changes in deep shape matching.

Another line of research [27, 45, 46, 57, 73, 80] (intrinsic methods) has focused on defining the convolution operator directly on the surface of the input shape. These methods are more suitable for deformable shapes and can leverage the structure of the surface encoded by the mesh, which is ignored by the extrinsic methods.

In previous works, intrinsic methods tend to perform better for non-rigid shape matching than extrinsic methods, with DiffusionNet [73] being considered the state-of-the-art feature extractor for shape matching. In this work, we will show that a simple modification to extrinsic feature extractors improves their overall performance for shape matching, making them comparable to or better than DiffusionNet.

Paper Organization The rest of the paper is structured as follows. In Sec. 3, we first introduce the deep functional map pipeline and our key definitions. We then state a theorem that shows that features learned in the deep functional map pipeline have a geometric interpretation under certain conditions. In light of this discovery, we introduce in Sec. 4 two modifications to the functional map pipeline, by promoting structural properties suggested by our analysis. Finally, in Sec. 5, we will show how these modifications improve overall performance in different shape matching scenarios.

3. Notation, Background & Motivation

3.1. Notation

Suppose we are given a pair of shapes S_1, S_2 represented as triangle meshes with respectively n_1 and n_2 ver-

tices. We compute the cotangent Laplace-Beltrami operator [49] of each shape S_i and collect the first k eigenfunctions as columns in a matrix denoted by Φ_i, and the corresponding eigenvalues in a diagonal matrix denoted as Δ_i. Φ_i is orthonormal with respect to the area (mass) matrix $\Phi_i^T A_i \Phi_i = \mathbb{I}$. We denote $\Phi_i^\dagger = \Phi_i^T A_i$, where $\cdot^\dagger$ is the (left) Moore–Penrose pseudo-inverse.

A pointwise map $T_{12} : S_1 \to S_2$ can be encoded in a matrix $\Pi_{12} \in \mathbb{R}^{n_1 \times n_2}$, where $\Pi_{12}(i, j) = 1$ if $T_{12}(i) = j$ and 0 otherwise. We will use the letter Π to denote pointwise maps in general. To every pointwise map, one can associate a functional map, simply via projection: $\mathbf{C}_{21} = \Phi_1^\dagger \Pi_{12} \Phi_2$.

Let $\mathcal{F}_\Theta$ be some feature extractor, which takes as input a shape and produces a set of feature functions (where Θ are some trainable parameters). We then have $F_1 = \mathcal{F}_\Theta(S_1)$. For simplicity we will omit explicitly stating $\mathcal{F}_\Theta$ and just denote the features associated with a shape S_i by F_i, where the presence of some feature extractor is assumed implicitly. Finally, $\mathbf{A}_i = \Phi_i^\dagger F_i$ denotes the matrix of coefficients of the feature functions in the corresponding reduced basis.

3.2. Deep Functional Map Pipeline

The standard Deep Functional Map pipeline can be described as follows.

Training: Pick a pair of shapes S_1, S_2 from some training set $\{S_i\}$, and compute their feature functions $F_1 = \mathcal{F}_\Theta(S_1), F_2 = \mathcal{F}_\Theta(S_2)$. Let $\mathbf{A}_1 = \Phi_1^\dagger F_1$, and $\mathbf{A}_2 = \Phi_2^\dagger F_2$ denote the coefficients of the feature functions in the reduced basis. Compute the functional map $\mathbf{C}_{12}$ by solving a least squares system inside the network:

$$\arg\min_{\mathbf{C}} \|\mathbf{C}\mathbf{A}_1 - \mathbf{A}_2\|_F^2. \tag{1}$$

This system can be further regularized by incorporating commutativity with the Laplacian [17]. A *training loss* is then imposed on this computed functional map (in the supervised setting) by comparing the computed functional map $\mathbf{C}_{12}$ with some ground truth:

$$\mathcal{L}_{sup}(\mathbf{C}_{12}) = \|\mathbf{C}_{12} - \mathbf{C}_{gt}\|_F^2. \tag{2}$$

Alternatively, (in the unsupervised setting) we can impose a loss on $\mathbf{C}_{12}$ by penalizing its deviation from some desirable properties, such as being an orthonormal matrix [69]. Finally, the loss above is used to back-propagate through the feature extractor network $\mathcal{F}_\Theta$ and optimize its parameters Θ.

Test time: Once the parameters of the feature extractor network are trained, we follow a similar pipeline at test time. Given a pair of *test shapes* S_1, S_2, we compute their feature functions $F_1 = \mathcal{F}_\Theta(S_1), F_2 = \mathcal{F}_\Theta(S_2)$. Let $\mathbf{A}_1 = \Phi_1^\dagger F_1$, and $\mathbf{A}_2 = \Phi_2^\dagger F_2$ denote their coefficients in the reduced basis. We then compute the functional map $\mathbf{C}_{12}$ by solving

the least squares system in Eq. (1). Given this functional map, a point-to-point (p2p) correspondence Π_{21} can be obtained by solving the following problem:

$$\Pi_{21} = \arg\min_{\Pi} \|\Pi\Phi_1 - \Phi_2 \mathbf{C}_{12}\|. \qquad (3)$$

This problem, which was proposed and justified in [22, 55] is row-separable and reduces to the nearest neighbor search between the rows of Φ_1 and the rows of $\Phi_2 \mathbf{C}_{12}$. This method of converting the fmap into a p2p map is referred to hereafter as the *adjoint method* following [55].

Advantages   This pipeline has several advantages: first, the training loss is imposed on the entire map rather than individual point correspondences, which has been shown to improve accuracy [17, 38]. Second, by using a reduced spectral basis, this approach strongly regularizes the learning problem and thus can be used even in the presence of limited training data [17, 73]. Finally, by accommodating both supervised and unsupervised losses, this pipeline allows to promote structural map properties without manipulating large and dense, e.g., geodesic distance, matrices.

Drawbacks & Motivation   At the same time, conceptually, the relation between the learned feature functions and the computed correspondences is still unclear. Indeed, although the feature (probe) functions are learned, they are used solely to formulate the optimization problem in Eq. (1). Moreover, the final pointwise correspondence in Eq. (3) is still computed from the Laplacian basis. Thus, it is not entirely clear what exactly is learned by the network $\mathcal{F}_{\Theta}$, and whether the learned feature functions can be used for any other task. Indeed, in practice, learned features can fail to highlight some specific repeatable regions on shapes, even if they tend to produce high-quality functional map matrices (Fig. 1).

3.3. Theoretical Analysis

In this subsection, we introduce some notions and state a theorem that will be helpful in our results below. We will use the same notation as in Sec. 3.1.

First, note that both during training (in Eq. (1)) and at test time (in Eq. (3)) the learned feature functions F_i, are used by projecting them onto the Laplacian basis. Thus, both from the perspective of the loss and also when computing the pointwise map, the deep functional map pipeline only uses the part of the functions that lies in the span of the Laplacian basis. Put differently, if we *modified* feature functions by setting $\tilde{F}_i = \Phi_i \Phi_i^\dagger F_i$ then both during training and at test time the behavior of the deep functional map pipeline will be identical when using either F_i or $\tilde{F}_i$.

Definition 3.1 (Complete feature functions). Motivated by this observation, we call the feature extractor F_{Θ} **complete**

if F_{Θ} produces feature (probe) functions that are contained in the corresponding LB eigenbasis. I.e.,

$$\mathcal{F}_{\Theta}(S_i) = \Phi_i \Phi_i^\dagger \mathcal{F}_{\Theta}(S_i), \text{ or equivalently} \qquad (4)$$
$$(Id - \Phi_i \Phi_i^\dagger)\mathcal{F}_{\Theta}(S_i) = 0. \qquad (5)$$

Definition 3.2 (Basis aligning functional maps). We also introduce a notion that relates to the properties of functional maps. Namely, given a pair of shapes, S_1, S_2 and a functional map $\mathbf{C}_{12}$, we will call it *basis-aligning* if the bases, when transformed by this functional map, align exactly. This can be summarized as follows: $\Phi_2 \mathbf{C}_{12} = \Pi_{21}\Phi_1$ for some point-to-point map Π_{21}.

For simplicity, for our result below we will also assume that all optimization problems have unique global minima. Thus, for the problem $\arg\min_{\mathbf{C}} \|\mathbf{C}\mathbf{A}_1 - \mathbf{A}_2\|$, this means that $\mathbf{A}_1$ must be full rank, whereas for the problem of type $\arg\min_{\Pi} \|\Pi F_1 - F_2\|$, this means that the rows of F_1 must be distinct (i.e., no two rows are identical, as vectors).

Theorem 3.1. Suppose the feature extractor F_{Θ} is *complete*. Let $\mathbf{A}_1 = \Phi_1^\dagger F_1$ and $\mathbf{A}_2 = \Phi_2^\dagger F_2$. Then, denoting $\mathbf{C}_{\text{opt}} = \arg\min_{\mathbf{C}} \|\mathbf{C}\mathbf{A}_1 - \mathbf{A}_2\|$, we have the following results hold:
(1) If $\Pi F_1 = F_2$ for some point-to-point map Π then $\mathbf{C}_{12} = \Phi_2^\dagger \Pi \Phi_1$ is basis-aligning. Moreover, $\mathbf{C}_{12} = \mathbf{C}_{\text{opt}}$, and extracting the pointwise map from $\mathbf{C}_{\text{opt}}$ via the adjoint method (see Eq. (3)) or via nearest neighbor search in the feature space $\min_{\Pi} \|\Pi F_1 - F_2\|$ will give the same result.
(2) Conversely, suppose that F_1, F_2 are complete and $\mathbf{C}_{\text{opt}}$ is basis aligning, then $\arg\min_{\Pi} \|\Pi F_1 - F_2\| = \arg\min_{\Pi} \|\Pi\Phi_1 - \Phi_2 C_{\text{opt}}\|$.

Proof. See the supplementary materials. □

Discussion: Note that in the theorem above, we used the notion of basis-aligning functional maps. Two questions arise: what are the necessary and sufficient conditions for a functional map to be basis-aligning?

A necessary condition is that the functional map must be *proper* as defined in [64]. Namely, a functional map $\mathbf{C}_{12}$ is **proper** if there exists a pointwise map Π_{21} such that $\mathbf{C}_{12} = \Phi_2^\dagger \Pi_{21}\Phi_1$. Note that for sufficiently high dimensionality k, *any* proper functional map must be basis aligning. This is because, since Φ_2 is guaranteed to be invertible in the full basis, there is a unique matrix $\mathbf{C}_{12}$ that will satisfy $\Pi_{21}\Phi_1 = \Phi_2 \mathbf{C}_{12}$ exactly.

Conversely, a *sufficient* condition for a functional map to be basis aligning is that the underlying point-to-point map is an isometry. Indeed, it is well known that isometries must be represented as block-diagonal functional maps, and moreover, isometries preserve eigenfunctions (see Theorem 2.4.1 in [54] and Theorem 1 in [70]). Thus, assuming that the functional map is of size k and that the kth and $(k+1)^{\text{st}}$

eigenfunctions are distinct, we must have that the functional map must be basis-aligning.

Finally, note that a functional map is basis-aligning if it is *proper* and if the image (by pull-back) of the k first eigenfunctions of the source lies in the range of the k first eigenfunctions of the target. This can therefore be considered as a measure of the *smoothness* of the map.

4. Proposed Modification

In the previous section, we provided a theoretical analysis of the conditions under which computed "probe" functions within the deep functional map pipeline can be used as pointwise descriptors directly, and lead to the same point-to-point maps as computed by the functional maps. In Sec. 5, we provide an extensive evaluation of using learned feature functions for pointwise map computation and thus affirm the validity of Theorem 3.1 in practice.

Our main observation is that the two approaches for point-to-point map computation are indeed often equivalent in practice, especially in "easy" cases, where existing state-of-the-art approaches lead to highly accurate maps. In contrast, we found that in more challenging cases, where existing methods fail, the two approaches are not equivalent and can lead to significantly different results.

Motivated by our analysis, we propose to use the structural properties suggested in Theorem 3.1 as a way to bridge this gap and improve the overall accuracy. Our proposed modifications are relatively simple, but they significantly improve the quality of computed correspondences, especially in "difficult" matching scenarios, as we demonstrate in Sec. 5.

The two key assumptions in Theorem 3.1 are *basis-aligning* functional maps and *complete* feature extractors. We propose modifying the functional map pipeline to satisfy the conditions of the theorem. Since the basis-aligning property is closely related to *properness*, we propose to impose that the predicted functional map to be proper, *i.e.* arises from some pointwise correspondence. For feature completeness, we suggest modifying the feature extractor to produce *smooth features*. We use the same notation as in Sec. 3.1.

4.1. Enforcing Properness

In this section, we propose two ways to enforce functional map properness and associated losses for both supervised and unsupervised training.

The adjoint method Given feature functions F_1, F_2, produced by a feature extractor, we compute the functional map $\mathbf{C}_{12-pred}$ as explained in Sec. 3. To compute a proper functional from it, we first convert $\mathbf{C}_{12-pred}$ into a p2p map $\Pi_{21-pred}$ in a differentiable way and then compute the "differentiable" proper functional map $\mathbf{C}_{21-proper} = \Phi_2^\dagger \Pi_{21-pred} \Phi_1$.
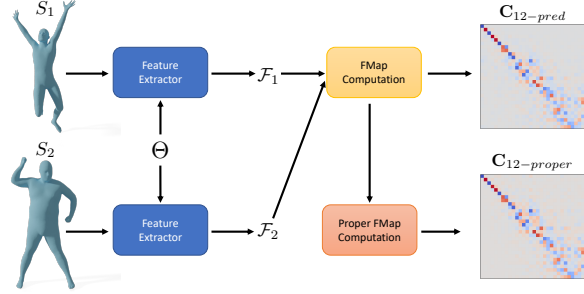

Figure 2. An overview of our revised deep functional map pipeline. The extracted features are used to compute the functional map and the proper functional map, as explained in Sec. 4.1

To compute $\Pi_{21-pred}$, denoting $G_1 = \Phi_1$ and $G_2 = \Phi_2 \mathbf{C}_{21-pred}$, we use:

$$\Pi_{21-pred}^{i,j} = \frac{\exp\left(\langle G_2^i, G_1^j \rangle / \tau\right)}{\sum_{k=1}^{n_1} \exp\left(\langle G_2^i, G_1^k \rangle / \tau\right)}. \qquad (6)$$

Here $\langle \cdot, \cdot \rangle$ is the scalar product measuring the similarity between G_1 and G_2, and τ is a temperature hyper-parameter. $\Pi_{21-pred}$ can be seen as a soft point-to-point map, formulated based on the adjoint conversion method described in [55], and computed in a differentiable manner, hence it can be used inside a neural network.

The feature-based method The feature-based method is similar to the adjoint method in spirit, the only difference being that $\Pi_{21-pred}$ is computed using the predicted features instead of the fmap. For this, we use Eq. (6), with $G_1 = F_1$ and $G_2 = F_2$. The modified deep functional map pipeline is illustrated in Fig. 2.

In addition to $C_{12-pred}$, the two previous methods allow to calculate $C_{21-proper}$. We adapt the functional map losses to take into account this modification.

In the supervised case, we modify the supervised loss (see Eq. (2)) by simply introducing an additional term into the loss:

$$\mathcal{L}_{proper} = \|\mathbf{C}_{12-pred} - \mathbf{C}_{12-proper}\|_F^2. \qquad (7)$$

The motivation behind this loss is that we want the predicted functional map to be as close as possible to the ground truth and stay within the space of proper functional maps.

In the unsupervised setting, we simply impose the standard unsupervised losses on the differentiable proper functional map $\mathbf{C}_{12-proper}$ instead of $\mathbf{C}_{12-pred}$. Specifically, in our experiments below, we use the following unsupervised losses:

$$\mathcal{L}_{unsup}(\mathbf{C}_{12}, \mathbf{C}_{21}) = \|\mathbf{C}_{12}\mathbf{C}_{21} - \mathbb{I}\|_F^2 + \|\mathbf{C}_{21}\mathbf{C}_{12} - \mathbb{I}\|_F^2$$
$$+ \|\mathbf{C}_{12}^\top\mathbf{C}_{12} - \mathbb{I}\|_F^2 + \|\mathbf{C}_{21}^\top\mathbf{C}_{21} - \mathbb{I}\|_F^2 \qquad (8)$$

4.2. As Smooth As Possible Feature Extractor

Another fundamental assumption of Theorem 3.1 is the completeness of the features produced by a neural network.

We have experimented with several ways to impose it and have found that it is not easy to satisfy it exactly in general because it would require the network to always produce features in some target subspace, which is not explicitly specified in advance. Moreover, we have found that explicitly projecting feature functions to a small reduced subspace can also hinder learning.

To circumvent this, we propose instead to *encourage* this property by promoting the feature extractor to produce smooth features.

The motivation for this is as follows. If F_i is complete, then there exist coefficients $a_1...a_k$ such that $F_i = \sum_{j=1}^{k} a_j \Phi_i^j$, where k is the size of the functional map used in Eq. (1). However, it's known that Fourier coefficients for smooth functions decay rapidly (faster than any polynomial, if f is of class C^l, the coefficients are $o(n^{-l})$), which means that the smoother the function is, the closer it will be to being complete for some index k.

Inspired by this, we propose the following simple modification to feature extractors used for deep functional maps. Since feature extractors are made of multiple layers, we propose to project the output of each layer into the Laplacian basis, diffuse it over the surface following [73], and then project it back to the ambient space before feeding it to the next layer, see Fig. 3. Concretely, for shape S, if f_i is the output of layer i, we feed to layer $i+1$ the function f_i', such that $f_i' = \Phi_j e^{-t\Delta} \Phi_j^\dagger f_i$, where Φ_j denotes the first j eigenfunctions of the Laplacian, Δ is a diagonal matrix containing the first j eigenvalues, and t is a learnable parameter. Please note there is no need to do this operation for the final layer, since the features will be projected into the Laplacian basis anyway, for computing the functional map. In practice, we observed that it is beneficial to set j to *be larger* than the size of the functional maps in Eq. (1). This allows the network to impose smoothness, while still allowing degrees of freedom to enable optimization.

Implementation details we provide implementation details, for all our experiments, in the supplementary. Our code and data will be released after publication.

5. Results & Applications

In this section, we provide results in a wide range of challenging tasks, showing the efficiency and applicability of our approach to different types of data and feature extractors.

Datasets Our method is evaluated on five datasets commonly used in literature, which include both human and animal datasets.
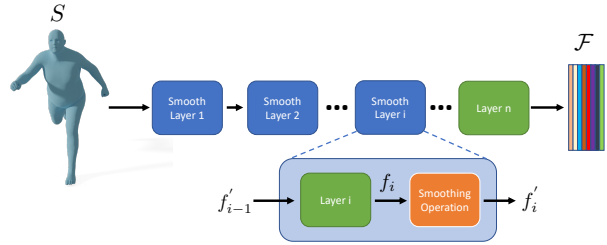

Figure 3. An overview of the feature extractor modification is shown here. The features are made smooth by projecting them into the Laplacian basis at the end of each layer.

To evaluate our method's performance for shape matching with humans, we use three datasets: FAUST [9], SCAPE [1], and SHREC'19 [47]. We utilize the remeshed versions of the first two datasets introduced in [65] and used in various follow-up works such as [17–20, 72, 73]. We follow the same train/test splits as in prior works. For unsupervised experiments, we utilize the datasets' oriented versions, as described in [72], denoted as **FA**, **SA**, and **SHA** for FAUST remeshed aligned, SCAPE remeshed aligned, and SHREC aligned, respectively.

We also evaluate our method on human segmentation using the dataset introduced in [45], which comprises segmented human models from various prior datasets. We use the same test split as in prior works.

For animal datasets, we use the SMAL-based dataset [43, 87] (denoted as **SMAL** hereafter), which consists of 50 organic, non-isometric, and non-rigid shapes represented as 3D meshes. We divide them into 25/25 shapes for training and testing, and to test the generalization capacity of our models, the animals and their positions during testing are never seen during training.

5.1. Non-Rigid Shape Matching

In this section, we evaluate our modification to the deep functional map pipeline in three different and challenging settings, including near-isometric supervised matching in Sec. 5.1.1, near-isometric unsupervised shape matching in Sec. 5.1.2, and finally, non-isometric non-rigid matching, both in supervised and unsupervised settings, in Sec. 5.1.3.

We evaluate our proposed modifications in the presence of three different feature extractors, an intrinsic feature extractor DiffusionNet [73] which is considered as the state of the art for shape matching, and two extrinsic feature extractors, DGCNN [79] and DeltaConv [81], which operate on point clouds. We will show that with our modifications, the latter feature extractors can surpass DiffusionNet in some scenarios.

Evaluation protocol Our evaluation protocol, following [33], measures the geodesic error between predicted maps

Model / Dataset	FR on FR		SR on SR		FR on SR		SR on FR		FR on SH		SR on SH	
	FM	NN	FM	NN	FM	NN	FM	NN	FM	NN	FM	NN
DiffusionNet	2.6	2.2	**2.9**	2.7	**3.4**	**3.1**	2.9	3.2	9.6	7.6	6.9	9.2
DiffusionNet - Ours	**2.6**	**2.0**	**2.9**	**2.4**	**3.4**	3.2	**2.7**	**2.3**	**5.7**	**5.7**	**5.0**	**5.6**
DGCNN	2.9	6.0	3.6	8.0	13.6	16.8	3.5	10.2	11.5	17.1	10.4	19.5
DGCNN - Ours	**2.6**	2.6	3.1	3.3	4.1	4.8	3.4	4.6	5.8	6.0	13.8	15.2
DeltaConv	2.7	13.8	4.4	21.2	18.1	26.6	12.2	24.4	19.4	32.6	27.7	34.9
DeltaConv - Ours	**2.6**	2.5	**2.9**	2.9	3.6	4.2	5.8	6.0	10.7	10.0	22.1	21.1

Table 1. Mean geodesic error of various feature extractors for supervised shape matching.

Figure 4. Qualitative results on SCAPE Remeshed dataset using DeltaConv [81], before and after our modifications.

Model / Dataset	FA on FA		SA on SA		FA on SA		SA on FA		FA on SHA		SA on SHA	
	FM	NN	FM	NN	FM	NN	FM	NN	FM	NN	FM	NN
DiffusionNet	3.9	6.5	4.5	6.5	5.4	8.5	3.7	6.0	6.1	11.9	6.0	10.6
DiffusionNet - Ours	**3.3**	**2.6**	**3.9**	**3.4**	**4.2**	**4.0**	**3.3**	**2.7**	6.2	**5.7**	**5.3**	**5.3**
DGCNN	3.9	9.3	5.0	11.2	7.0	13.6	4.1	11.9	6.7	17.1	6.5	16.7
DGCNN - Ours	3.9	2.8	4.6	3.8	5.2	5.0	4.0	3.4	6.5	**5.7**	6.2	6.0
DeltaConv	3.8	12.9	4.7	15.5	5.1	17.4	4.0	16.5	7.0	23.6	6.7	25.0
DeltaConv - Ours	3.6	3.5	4.4	4.0	4.7	4.7	4.0	3.5	**6.0**	6.1	6.7	7.7

Table 2. Mean geodesic error of various feature extractors for unsupervised shape matching.

and the given ground truth, normalized by the square root of the total surface area. All reported values are multiplied by $\times 100$ for clarity.

In our experiments, we use the notation "X on Y" to indicate training on X and testing on Y. We use **FM** to represent the point-to-point map derived from the functional map and **NN** for the point-to-point map extracted by nearest neighbor in the feature space. We use "Feature Extractor - Ours" to denote our training approach for DGCNN and DeltaConv, which involves training the network with both smoothness and properness enforced, while for DiffusionNet, we only enforce properness since smoothness is enforced by construction. We report properness results only for the feature-based method and include adjoint method results in the supplementary materials.

5.1.1 Supervised Shape Matching

In this section, we test the quality of our proposed modifications in the supervised setting. For this, we follow the same pipeline described in Sec. 3, and train our network with the supervised loss in Eq. (7). Also, to stress the generalization power of each network, we do multiple tests, by training on one dataset, and testing on another one, following multiple previous works [17, 20, 72]. For this, we use the **FR**, **SR**, and **SH** datasets. The results are summarized in Tab. 1.

First of all, we can see that our approach improves the global result for all feature extractors, on all datasets. The performance increase can be up to 80%, as is the case for DeltaConv and DGCNN on the scenario **FR on SR**, where *e.g.* the error has decreased from 18.1 to 3.6.

Another interesting result to note is that using our modifications, the extrinsic feature extractors, which were failing in the generalization scenarios, now perform as well as DiffusionNet. Finally, and perhaps more *importantly*, after our modifications, the results obtained with the nearest neighbor are much better, and get closer to the maps extracted by converting the functional map, demonstrating, for the first time, that the features learned in the deep fmap framework do have a geometric meaning, and can be used directly for matching, even without the need to compute a functional map. We explain the slight discrepancy between the result obtained with fmap and NN by the fact that, in practice, the conditions of the theorem are not 100% satisfied, since
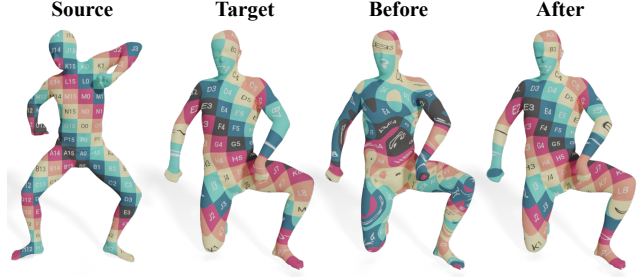
we only used approximations to obtain them. In Fig. 4, we show the quality of the produced maps, before and after our modifications. It can be seen that our modifications produce visually plausible correspondences.

5.1.2 Unsupervised Shape Matching

In this section, we test the quality of our modifications in the case of unsupervised learning on near-isometric data. We follow the same method as explained in Sec. 3 with the loss in Eq. (8), and use three feature extractors: DGCNN, DiffusionNet and DeltaConv, and three datasets: **FA**, **SA** and **SHA**. The results are summarized in Tab. 2.

As for the supervised case, our modifications improve the results for all scenarios and for all feature extractors. For example, the result improved by over 70 % for DeltaConv on the **FRA** on **SHA** scenario.

In addition, the extrinsic feature extractors are now on par with DiffusionNet, and the NN results are now close to those obtained with fmap, reinforcing the conclusions obtained in the supervised case Sec. 5.1.1.

Note that the effect of enforcing properness is more visible in this context. This can be seen for example in the case of DiffusionNet in Table Sec. 5.1.1 since for this feature extractor the only change brought about by our method is via properness. In fact, in the supervised setting, since the training is done using the ground truth-functional maps, and these are proper, this forced the network to learn features that produce functional maps that are as proper as possible, whereas in the unsupervised setting, the network is trained with losses that impose structural properties on the functional maps such as orthogonality, and no properness is involved.

Animals Dataset	Supervised		Unsupervised	
	FM	NN	FM	NN
DiffusionNet	6.1	8.7	8.0	16.6
DiffusionNet - Ours	5.7	7.8	6.4	10.1
DGCNN	8.3	18.4	10.9	19.1
DGCNN - Ours	<u>5.0</u>	<u>4.8</u>	<u>5.3</u>	**5.5**
DeltaConv	5.4	20.7	9.5	24.1
DeltaConv - Ours	**4.7**	**4.2**	**5.1**	<u>5.6</u>

Table 3. Mean geodesic error comparison of different feature extractors on the **SMAL** dataset, using supervised and unsupervised methods. The highest performing result in each column is denoted in **bold**, while the second best is denoted with <u>underline</u>.

We can see that after imposing properness, both functional map and NN results improve, while also getting closer to each other.

5.1.3 Non-Isometric Shape Matching

In this section, we test the utility of our modifications in the case of non-rigid non-isometric shape matching, on the **SMAL** dataset. We test it both in the case of supervised and unsupervised learning, following the same procedure as in Sec. 5.1.1 and Sec. 5.1.2. The results are summarized in Tab. 3. As in the previous sections, our modifications improve the results for all feature extractors for both supervised and unsupervised matching. For example, the results for supervised DeltaConv improve by approximately 80%, from 20.7 to 4.2.

More surprisingly, using our modifications, in the non-isometric setting, DiffusionNet is surpassed as the best feature extractor for shape matching, as DGCNN and DeltaConv achieve better results, using either the functional map or NN method to extract the p2p map.

5.2. Generalization Power of Geometric Features

We conclude our experiments by evaluating the generalization ability of the learned features in the deep functional map pipeline. Our goal is to investigate whether the extracted features contain sufficient information to be used for other tasks without the need for significant fine-tuning.

To conduct this experiment, we employ the human segmentation dataset presented in [45]. We first extract features from this dataset using a network that has been pre-trained on shape matching. We use the pre-trained DiffusionNet on the **SA** dataset with the unsupervised. Next, we refine the features for the segmentation task using only a small fraction of the training data. To accomplish this, we utilize a point-wise MLP to avoid complex convolution and pooling operations and to provide an accurate measure of the feature quality.

In Fig. 5, we plot the accuracy of the MLP as a function of the number of finetuning shapes. First note that the accuracy of the results improves with the number of training
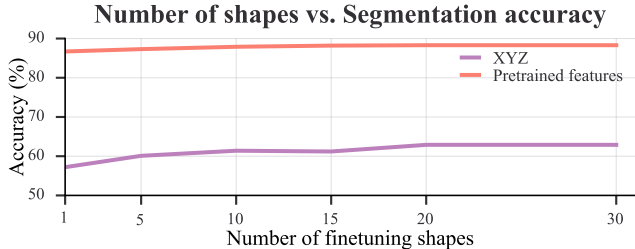

Figure 5. Evolution of human segmentation accuracy as a function of the number of shapes used for the pointwise MLP training.

shapes (up to a certain point, where most likely the limited capacity of pointwise MLP is reached). Note also that our features, which are pre-trained in a completely unsupervised manner, produce *significantly* higher accuracy than the raw XYZ coordinates. We attribute this feature utility in a downstream task to the fact that these features capture the geometric structure of the shapes in a compact and invariant manner. To further demonstrate the generalization power of the pretrained features, we conducted a similar experiment on the RNA dataset [58] and present the results in the supplementary material.

Ablation Studies In the supplementary, we report an ablation study that demonstrates the effectiveness of the individual components we introduced, in addition to a comparison of our modified pipeline to other recent baseline methods for shape matching.

6. Conclusion & Limitations

In this work, we studied the information contained in the features learned in the deep functional map pipelines, an aspect that has been neglected by all previous works. For this, we established a theorem according to which, under certain mild conditions, these features allow for *direct* computation of point-to-point maps. Inspired by our analysis, we introduced two simple modifications to the deep functional map pipeline. We have shown that applying these modifications improves the overall accuracy of the pipeline in different scenarios, including supervised and unsupervised matching of non-rigid shapes, and makes the features usable for other downstream tasks.

A limitation of our approach is that it is currently geared towards *complete* shape correspondence and must be adapted for partial shape matching. Moreover, it will be interesting to investigate how feature smoothness can be imposed in the context of noisy point clouds or other domains such as graphs or images.

Supplementary Materials for:
Understanding and Improving Features Learned in Deep Functional Maps

Souhaib Attaiki   Maks Ovsjanikov

LIX, École Polytechnique, IP Paris

In this document, we collect all the results and discussions, which, due to the page limit, could not find space in the main manuscript.

More precisely, we first provide the proof of the theorem we introduced in Sec. 3.3 of the main text in Appendix A. Then, the details of the implementation are provided in Appendix B. We present a more in-depth analysis of the modifications we introduced to the deep functional map pipeline in Appendix C. A comparison with other shape-matching methods is provided in Appendix D, as well as additional results regarding the generalization power of pre-trained features in Appendix E. Finally, some qualitative results are included in Appendix F.

A. Proof of Theorem 3.1

In Sec. 3.3 of the main text, we stated a theorem that shows that the maps obtained with the adjoint method, or the nearest neighbor in the feature space are equivalent under some conditions. In this section, we formally restate it and provide proof.

As mentioned in the main body, in our result below we assume that all optimization problems have unique global minima. Thus, for the problem $\arg\min_{\mathbf{C}} \|\mathbf{C}\mathbf{A}_1 - \mathbf{A}_2\|$, this means that $\mathbf{A}_1$ must be full row rank, whereas, for the problem of type $\arg\min_{\Pi} \|\Pi F_1 - F_2\|$, this means that the rows of F_1 must be distinct (i.e., no two rows are identical, as vectors).

Theorem 3.1. Suppose the feature extractor F_Θ is *complete*. Let $\mathbf{A}_1 = \Phi_1^+ F_1$ and $\mathbf{A}_2 = \Phi_2^+ F_2$. Then, denoting $\mathbf{C}_{\mathrm{opt}} = \arg\min_{\mathbf{C}} \|\mathbf{C}\mathbf{A}_1 - \mathbf{A}_2\|$, we have the following results hold:

(1) If $\Pi F_1 = F_2$ for some point-to-point map Π then $\mathbf{C}_{12} = \Phi_2^+ \Pi \Phi_1$ is basis-aligning. Moreover, $\mathbf{C}_{12} = \mathbf{C}_{\mathrm{opt}}$ and extracting the pointwise map from $\mathbf{C}_{\mathrm{opt}}$ via the adjoint method, or via nearest neighbor search in the feature space $\min_{\Pi} \|\Pi F_1 - F_2\|$ will give the same result.

(2) Conversely, suppose that $\mathbf{C}_{\mathrm{opt}}$ is basis aligning, then $\arg\min_{\Pi} \|\Pi F_1 - F_2\| = \arg\min_{\Pi} \|\Pi \Phi_1 - \Phi_2 C_{\mathrm{opt}}\|$.

Proof. (1) If $\Pi F_1 = F_2$ and F_1, F_2 are complete by assumption then we have $F_1 = \Phi_1 \mathbf{A}_1$ and $F_2 = \Phi_2 \mathbf{A}_2$ so that:

$$\Pi \Phi_1 \mathbf{A}_1 = \Phi_2 \mathbf{A}_2 \qquad (9)$$

Setting $\mathbf{C}_{12} = \Phi_2^+ \Pi \Phi_1$ and pre-multiplying Eq. (9) by Φ_2^+ we obtain $\mathbf{C}_{12}\mathbf{A}_1 = \mathbf{A}_2$. Thus, $\|\mathbf{C}_{12}\mathbf{A}_1 - \mathbf{A}_2\| = 0$,

and it follows that $\mathbf{C}_{\mathrm{opt}} = \mathbf{C}_{12}$, since $\mathbf{A}_1$ assumed to be of full row rank (and thus $\arg\min_{\mathbf{C}} \|\mathbf{C}\mathbf{A}_1 - \mathbf{A}_2\|$ has a unique optimum).

Moreover, using $\mathbf{C}_{12}\mathbf{A}_1 = \mathbf{A}_2$, we get $\Phi_2 \mathbf{A}_2 = \Phi_2 \mathbf{C}_{12}\mathbf{A}_1$. Combining this with Eq. (9), we get $\Pi \Phi_1 \mathbf{A}_1 = \Phi_2 \mathbf{C}_{12}\mathbf{A}_1$. Using the fact that $\mathbf{A}_1$ is full rank, this implies that $\Pi \Phi_1 = \Phi_2 \mathbf{C}_{12}$, and thus $\mathbf{C}12$ is basis-aligning.

Finally, we note that since the same pointwise map satisfies $\|\Pi F_1 - F_2\| = \|\Pi \Phi_1 - \Phi_2 \mathbf{C}_{12}\| = 0$, minimizing both energies with respect to Π would result in the same map.

(2) By assumption $\mathbf{C}_{\mathrm{opt}}$ is basis-aligning. Thus, $\Pi_{21}\Phi_1 = \Phi_2 \mathbf{C}_{\mathrm{opt}}$ for some pointwise map Π_{21}. Thus,

$$\min_{\Pi} \|\Pi \Phi_1 - \Phi_2 \mathbf{C}_{\mathrm{opt}}\| = \Pi_{21} \qquad (10)$$

Now let's consider the problem

$$\min_{\Pi} \|\Pi F_1 - F_2\| = \min_{\Pi} \|\Pi F_1 - F_2\|^2 \qquad (11)$$

The objective can be decomposed into two parts, one that lies within the span of Φ_2 and the one outside of it, as follows:

$$
\begin{aligned}
E(\Pi) =& \|\Pi F_1 - F_2\|^2 \\
=& \|\Phi_2^+ (\Pi F_1 - F_2)\|^2 + \|(I - \Phi_2\Phi_2^+)(\Pi F_1 - F_2)\|^2 \\
=& E_1(\Pi) + E_2(\Pi)
\end{aligned}
$$

Using the fact that F_i are complete, we have $F_i = \Phi_i \mathbf{A}_i$. It follows that $E_1 = \|\Phi_2^+ \Pi \Phi_1 \mathbf{A}_1 - \mathbf{A}_2\|^2$.

On the other hand, using the fact that F_2 is complete, we have $(I - \Phi_2\Phi_2^+)F_2 = 0$, and thus $E_2 = \|(I - \Phi_2\Phi_2^+)\Pi F_1\|^2$.

Now recall that by assumption $\mathbf{C}_{\mathrm{opt}}$ is basis-aligning and thus there exists some pointwise map Π_{21} such that $\Pi_{21}\Phi_1 = \Phi_2 \mathbf{C}_{\mathrm{opt}}$.

Consider $E_1(\Pi)$ for an arbitrary pointwise map Π. We have:

$$
\begin{aligned}
\min_{\mathbf{C}} \|\mathbf{C}\mathbf{A}_1 - \mathbf{A}_2\| =& \|C_{\mathrm{opt}}\mathbf{A}_1 - \mathbf{A}_2\| \\
=& \|\Phi_2^+ \Pi_{21}\Phi_1 \mathbf{A}_1 - \mathbf{A}_2\| \\
=& E_1(\Pi_{21})
\end{aligned}
$$

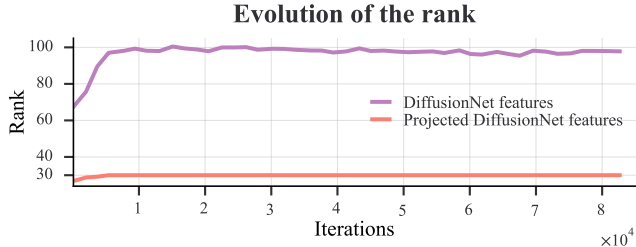It follows that we must have that $E_1(\Pi_{21}) \le E_1(\Pi)$ for any pointwise map Π.

Figure 6. Evolution of the rank of features produced by DiffusionNet, as well as their projection in the Laplacian-Basis, during training.

Moreover observe that for Π_{21} we have:

$$
\begin{aligned}
E_2(\Pi_{21}) &= \|(I - \Phi_2\Phi_2^+)\Pi_{21}F_1\|^2 \\
&= \|(I - \Phi_2^+\Phi_2^+)\Pi_{21}\Phi_1\mathbf{A}_1\|^2 \\
&= \|(I - \Phi_2\Phi_2^+)\Phi_2\mathbf{C}_{\text{opt}}\mathbf{A}_1\|^2 \\
&= \|\Phi_2\mathbf{C}_{\text{opt}}\mathbf{A}_1 - \Phi_2\mathbf{C}_{\text{opt}}\mathbf{A}_1\| \\
&= 0
\end{aligned}
$$

Thus, for an arbitrary pointwise map Π we must have $E_2(\Pi_{12}) \leq E_2(\Pi)$.

It therefore follows that $\arg\min_\Pi \|\Pi F_1 - F_2\| = \arg\min_\Pi (E_1(\Pi) + E_2(\Pi)) = \Pi_{21}$, and thus $\arg\min_\Pi \|\Pi F_1 - F_2\| = \arg\min_\Pi \|\Pi\Phi_1 - \Phi_2 C_{\text{opt}}\|$. $\square$

Verification of the assumptions of the theorem Note that as mentioned above, we first assumed that $\mathbf{A}_1$ is of full rank, and second, the rows of F_1 must be distinct so that all optimization problems have unique minima. We would like to point out that these assumptions are very weak and easily hold in practice.

Indeed, as far as the second condition is concerned, it is easily verifiable because of numerical precision, as it is very unlikely that two distinct points are assigned the exact same feature vectors. In practice, for example, we compute for each point in a feature produced by DiffusionNet, the distance to its nearest neighbor, and we take the average over the whole shape. We find that this distance is equal to 0.15, while for reference, it is equal to 0.0004 for the XYZ coordinates, which shows that this assumption is justified.

For the first assumption, we plot in Fig. 6 the evolution of the rank of the features produced by DiffusionNet during training, as well as the evolution of the rank of their projection on the spectral basis (*i.e.* $\mathbf{A}_i$). We can see that throughout the training, the rank of the projected features is always equal to 30, which is the same dimension as the spectral basis used in all our experiments, which proves that the assumption of the full rank of $\mathbf{A}_1$ is verified.

Impact of the proposed modifications on the conditions of the theorem In this paragraph, we provide measures

Property	DGCNN		DeltaConv	
	before	after	before	after
Basis alignment ($\downarrow$)	9.7	**4.8**	11.3	**5.9**
Properness ($\downarrow$)	9.6	**4.4**	13.1	**7.9**
Completeness ($\uparrow$)	69 %	**90 %**	64 %	**83 %**

Table 4. Impact of our modifications on the key metrics.

of the individual conditions of the theorem before and after applying our modifications in order to shed light on their effectiveness.

Given a functional map $\mathbf{C}_{12}$ predicted by the fmap framework, we compute its basis aligning property by computing the Chamfer distance between $\Phi_2\mathbf{C}_{12}$ and Φ_1 using the notation in Definition 3.2 of the main text (note that $\|\Phi_2\mathbf{C}_{12} - \Pi_{21}\Phi_1\|$ is a one-way Chamfer distance). For properness, given a predicted functional map $\mathbf{C}$, we compute its proper version $\mathbf{C}_{\text{proper}}$ using the adjoint method and measure properness by computing $\|\mathbf{C} - \mathbf{C}_{\text{proper}}\|_2^2$. For completeness, using the same notation as in Eq. (4), we measure it using $1 - \frac{\|\mathcal{F}_\Theta(S_i) - \Phi_i\Phi_i^\dagger\mathcal{F}_\Theta(S_i)\|_2^2}{\|\mathcal{F}_\Theta(S_i)\|_2^2}$, this quantity is between 0 and 1 (higher is better).

We report these measures in Tab. 4 for the same setup used in our ablation study, i.e., unsupervised shape matching on "FA on FA" (see Appendix C), for two main backbones: DGCNN and DeltaConv (note that in DiffusionNet completeness is enforced by construction). It can be seen that our modifications improve the measured properties.

B. Implementation Details

In all our experiments with functional maps, we used the functional map size of $k = 30$. For the Laplace-Beltrami computation, we used the discretization introduced in [74].

In our experiments in Sec. 5.2 of the main text, we used three different networks, DGCNN [79], DiffusionNet [73] and DeltaConv [81]. For this, we used the publicly available implementations [1] [2] [3] released by the authors. In all our experiments, we used the default segmentation configuration provided by the authors in their respective papers, with an output dimension of 128. For all these experiments, we used the Adam optimizer [34], with a learning rate of 0.001. We used data augmentation in all our experiments. In particular, we augment the training data on the fly by randomly rotating the input shapes, varying the position of each point by Gaussian noise, and applying random scaling in the interval [0.9, 1.1].

For the proper map computation (Eq. 6 of the main text), we used $\tau = 0.07$.

---

[1] https://github.com/WangYueFt/dgcnn
[2] https://github.com/nmwsharp/diffusion-net
[3] https://github.com/rubenwiersma/deltaconv

Model / Dataset	FA on FA		SA on SA		FA on SA		SA on FA		FA on SHA		SA on SHA	
	FM	NN	FM	NN	FM	NN	FM	NN	FM	NN	FM	NN
DiffusionNet - ⓪	3.9	6.5	4.5	6.5	5.4	8.5	3.7	6.0	6.1	11.9	6.0	10.6
DiffusionNet - ①	3.4	3.2	4.2	4.4	4.3	4.9	3.4	3.7	**5.5**	5.8	**5.3**	6.1
DiffusionNet - ②	**3.3**	**2.6**	**3.9**	**3.4**	**4.2**	**4.0**	**3.3**	**2.7**	6.2	**5.7**	**5.3**	**5.3**
DGCNN - ⓪	3.9	9.3	5.0	11.2	7.0	13.6	4.1	11.9	6.7	17.1	6.5	16.7
DGCNN - ①	**3.5**	5.6	4.4	7.3	7.1	11.5	3.6	8.8	7.0	13.2	5.7	13.0
DGCNN - ②	3.7	3.5	4.8	5.2	7.4	11.3	4.0	5.1	8.0	10.7	6.3	8.4
DGCNN - ③	3.9	4.7	4.8	5.7	5.0	6.2	3.9	5.2	6.5	7.4	6.4	8.4
DGCNN - ① + ③	**3.5**	3.6	**4.3**	4.4	**4.5**	5.2	**3.4**	4.5	**5.4**	5.8	**5.3**	6.5
DGCNN - ② + ③	3.9	**2.8**	4.6	**3.8**	5.2	**5.0**	4.0	**3.4**	6.5	**5.7**	6.2	**6.0**
DeltaConv - ⓪	3.8	12.9	4.7	15.5	5.1	17.4	4.0	16.5	7.0	23.6	6.7	25.0
DeltaConv - ①	**3.4**	7.0	4.2	9.5	4.5	13.4	3.6	12.4	5.8	18.0	6.2	19.2
DeltaConv - ②	3.6	**3.5**	4.3	5.1	5.9	8.7	3.9	7.0	7.0	10.6	6.2	11.5
DeltaConv - ③	3.8	5.7	4.5	6.7	4.7	7.1	3.8	5.8	6.3	9.1	6.6	11.7
DeltaConv - ① + ③	**3.4**	3.6	**4.1**	4.6	**4.3**	5.2	**3.4**	4.2	**5.3**	6.3	**5.4**	**7.1**
DeltaConv - ② + ③	3.6	**3.5**	4.4	**4.0**	4.7	**4.7**	4.0	**3.5**	6.0	**6.1**	6.7	7.7

Table 5. Ablation Study on the components of our method. ⓪: no modification. ①: use properness based on the adjoint method. ②: use properness using the feature-based method. ③: make the feature extractor as smooth as possible using the method introduced in Sec. 4.2 of the main text. We highlight the best result of each feature extractor in **bold**.

To make the feature extractor smooth, we project the features onto a Laplace-Beltrami basis of size $j = 128$, which is the same size used in the original DiffusionNet implementation.

Regarding the experiment in Sec. 5.3 of the main text, we used the pre-trained DiffusionNet model on the Scape Remeshed Aligned dataset, with unsupervised and properness losses. The downstream point MLP consists of 4 layers, and we train it with the Adam optimizer for 300 iterations, using a learning rate of 0.001. The training shape is randomly sampled from the training data set.

C. Ablation Study

In Sec. 4 of the main text, we introduced two modifications to the functional map pipeline, namely imposing properness on the functional map and requiring the features to be as smooth as possible. Here we show the effect of each modification independently. For illustration, we will use the unsupervised near-isometric matching experiment (Sec. 5.2.2 of the main text), but a similar conclusion can be drawn in all other scenarios.

The results are summarized in Tab. 5. In this table, we show the result for each architecture without modification ⓪, with properness with the adjoint method ①, with feature-based properness ②, using the smoothness operation ③ or the combination of the latter. Since in DiffusionNet, the smoothing operation is performed by construction, we only include the results with properness.

We can see that each of our modifications improves the result of the vanilla feature extractor, and for optimal performance, both modifications should be used. In particular, we noticed that imposing the properness using the feature-based method gives a slightly better result, so we advocate using this method.

Model / Dataset	FR on FR	SR on SR	FR on SR	SR on FR	FR on SH	SR on SH
3D-CODED	2.5	31.0	31.0	33.0	–	–
FMNet	11.0	30.0	30.0	33.0	–	–
HSN	3.3	3.5	25.4	16.7	–	–
GeomFmaps	3.1	4.4	11.0	6.0	9.9	12.2
TransMatch	2.7	18.3	33.6	18.6	21.0	38.8
GeomFmaps + DiffusionNet	2.6	2.9	3.4	2.9	9.6	6.9
SRFeat	**1.1**	2.2	3.9	2.5	9.9	6.2
GeomFmaps + DiffusionNet - Ours	2.0	2.4	<u>3.2</u>	<u>2.3</u>	**5.7**	<u>5.6</u>
SRFeat - Ours	<u>1.3</u>	**1.8**	**2.9**	**1.8**	<u>5.8</u>	**5.4**

Table 6. Accuracy of various supervised shape matching methods for near-isometric shape matching. Our modifications achieve state-of-the-art results. The best result in each column is highlighted in **bold**, and the second best is highlighted using <u>underline</u>.

As we explained in the main text, the difference between the result obtained with the functional map, and the result with the nearest neighbor (NN) method is explained by the fact that the conditions of the theorem are not fully satisfied. Indeed, while the results of the two approaches are generally very close, after our modifications, there is variability in terms of which method produces the best results depending on the dataset. Remark that if the NN approach is better, this suggests that higher frequencies in the learned feature functions are beneficial for correspondence. However, the opposite can (and indeed does) occur, in that those high frequencies can hinder results since they are not penalized during training (as the functional map losses are computed after projecting the features onto a low-frequency basis).

D. Comparison with other methods

In this section, we compare our method to other recent shape-matching methods and evaluate the effect of our proposed modifications on additional baseline approaches. We compare our method in supervised and unsupervised near-isometric shape-matching experiments in Sec. 5.2.1 and Sec. 5.2.2 of the main text, respectively.

Note that since our proposed modifications are general and can be applied to *any deep functional map pipeline* in principle, we also tested their effects on additional methods. In addition to the comparisons shown in the main manuscript, below we also test our modifications on two very recent approaches: SRFeat [37] for the supervised case, and DUO-FMap [16] for the unsupervised case. Because both SRFeat and DUO-FMap use DiffusionNet as a backbone, we only enforce the properness using the feature-based method, following Sec. 4.1 of the main text.

We use the same protocol as in the main text. Namely, the geodesic error is normalized by the square root of the total surface area, values are multiplied by $\times 100$ for clarity, and the notation "X on Y" means that we train on X and test on Y. We denote the methods using our proposed modifications by "Method - Ours".

Concerning the supervised near-isometric shape-matching experiment, we compare to FMNet [38], 3DCODED [28], HSN [80], TransMatch [78], Ge-

Model / Dataset	FA on FA	SA on SA	FA on SA	SA on FA	FA on SHA	SA on SHA
SURFMNet	15.0	12.0	30.0	30.0	–	–
UnsupFMNet	10	16.0	29.0	22.0	–	–
WSupFMNet	3.3	7.3	11.7	6.2	–	–
DeepShells	**1.7**	_2.5_	5.4	2.7	26.3	22.8
NeuroMorph	8.5	29.9	28.5	18.2	26.3	27.6
WSupFMNet + DiffusionNet	3.9	4.5	5.4	3.7	6.1	_6.0_
DUO-FMap	2.5	2.6	4.2	_2.7_	6.4	8.4
WSupFMNet + DiffusionNet - Ours	2.6	3.4	_4.0_	_2.7_	_5.7_	5.3
DUO-FMap - Ours	_2.3_	**2.4**	**3.0**	**2.4**	5.5	5.3

Table 7. Accuracy of various unsupervised shape matching methods for near-isometric shape matching. Our modifications achieve state-of-the-art results. The best result in each column is highlighted in **bold**, and the second best is highlighted using underline.

omFMaps [17], and SRFeat [37]. Results are summarized in Tab. 6. It can be seen that our method achieves state-of-the-art results among supervised methods, especially in challenging cases such as testing on the SHREC Remeshed dataset.

For the unsupervised setting, we test our method against SURFMNet [69], UnsupFMNet [31], WSupFMNet [72], Deep Shells [20], Neuromorph [19], and DUO-FMap [16]. Results are summarized in Tab. 7. As can be seen, our method achieves state-of-the-art results in this scenario also, especially in challenging cases involving generalization, where all competing methods fail. It can also be seen that the modifications we propose are complementary to the different versions of the deep functional map pipeline. Remarkably, our method brings consistent and significant improvements throughout *all cases* and baseline approaches that we tested.

We would like to emphasize that our main contribution is both an analysis and a set of *improvements* for the deep functional map pipeline in general. As such, rather than a particularly new approach for correspondence, our key contribution is a set of modifications, which can be adapted within different deep functional map pipelines. We emphasize this because our approach is flexible and, as illustrated in the results, can be beneficial for different methods proposed in the literature for both supervised and unsupervised cases.

E. Generalization Power of Pre-Trained Features

In Sec. 5.3 of the main text, we showed the generalization power of our pretrained features on the task of human segmentation. Here we show more results consolidating the fact that learned features do have a geometric signification.

For this, we tested the utility of the pre-trained features for the task of molecular surface RNA segmentation. We used the RNA dataset introduced in [58], composed of 640 RNA triangle meshes, where each vertex is labeled into one of 259 atomic categories. We used the same 80/20% split for training and test sets as in previous works.

We follow the same setup as in Sec. 5.3 of the main text. Specifically, we train a DiffusionNet network for shape

Method	Accuracy
SplineCNN [23]	53.6 %
SPHNet [58]	80.2 %
SurfaceNetworks [35]	88.5%
DiffusionNet - XYZ	90.5%
DiffusionNet - HKS	90.6%
DiffusionNet - Pretrained features	**90.8%**

Table 8. Accuracy of various methods for RNA segmentation.
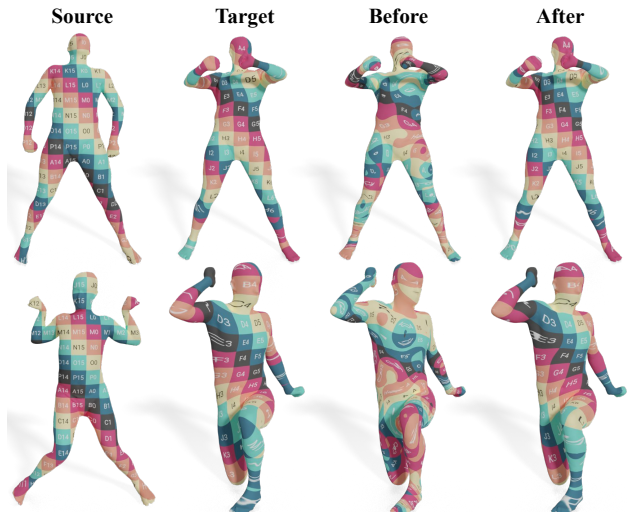

Figure 7. Qualitative results on SCAPE Remeshed dataset using DeltaConv, before and after using the modifications we have introduced.

matching in an unsupervised manner on the RNA data. We then extract features for each shape using the trained network. Finally, these features are used to train another DiffusionNet for the semantic segmentation task in a supervised manner.

The results are summarized in Tab. 8. We can see that our pre-trained features outperform XYZ and HKS [76] (note that due to the relatively large size of the training set, the improvement starts to saturate). We take this as further evidence that the features extracted using our approach encode geometric information that can be useful in various shape analysis tasks.

F. Qualitative Results

In this section, we present some qualitative results with our method.

In Fig. 7, we show the quality of the produced maps, before and after our modifications. It can be seen that our modifications produce visually plausible correspondences.

In Fig. 8, we show how repeatable the features are, before and after our modifications, the intensity is color coded.

Figure 8. Visualization of the intensity of the same feature learned by DeltaConv for shape matching, on several shapes, before and after using our modifications. The intensity is color coded.

We can see that before the modifications, the features are mostly flat and non-distinctive, making them not useful for matching using the nearest neighbor method, or for use in a downstream task, whereas, after our modifications, we can see that the features are activated over the same region, over multiple shapes, and they are not flat since they vary according to the geometry.

References

[1] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. SCAPE. *ACM Transactions on Graphics*, 24(3):408–416, July 2005. 6

[2] Souhaib Attaiki, Lei Li, and Maks Ovsjanikov. Generalizable local feature pre-training for deformable shape analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023. 2

[3] Souhaib Attaiki and Maks Ovsjanikov. NCP: Neural correspondence prior for effective unsupervised shape matching. In *Advances in Neural Information Processing Systems*, 2022. 2

[4] Souhaib Attaiki, Gautam Pai, and Maks Ovsjanikov. DPFM: Deep partial functional maps. In *2021 International Conference on 3D Vision (3DV)*. IEEE, Dec. 2021. 1, 2

[5] Matan Atzmon, Haggai Maron, and Yaron Lipman. Point convolutional neural networks by extension operators. *ACM Transactions on Graphics*, 37(4):1–12, Aug 2018. 3

[6] Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *2011 IEEE international conference on computer vision workshops (ICCV workshops)*, pages 1626–1633. IEEE, 2011. 1, 2

[7] Xuyang Bai, Zixin Luo, Lei Zhou, Hongkai Chen, Lei Li, Zeyu Hu, Hongbo Fu, and Chiew-Lan Tai. PointDSC: Robust point cloud registration using deep spatial consistency. In *CVPR*, 2021. 1

[8] Ilya Baran, Daniel Vlasic, Eitan Grinspun, and Jovan Popović. Semantic deformation transfer. *ACM Transactions on Graphics*, 28(3):1–6, July 2009. 1

[9] Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. FAUST: Dataset and evaluation for 3d mesh registration. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, June 2014. 1, 6

[10] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. 2

[11] Oliver Burghard, Alexander Dieckmann, and Reinhard Klein. Embedding shapes with Green's functions for global shape matching. *Computers & Graphics*, 68:1–10, 2017. 2

[12] Wenming Cao, Zhiyue Yan, Zhiquan He, and Zhihai He. A comprehensive survey on geometric deep learning. *IEEE Access*, 8:35929–35949, 2020. 2

[13] Luca Cosmo, Emanuele Rodola, Michael M Bronstein, Andrea Torsello, Daniel Cremers, and Y Sahillioglu. Shrec'16: Partial matching of deformable shapes. *Proc. 3DOR*, 2(9):12, 2016. 1, 2

[14] Haowen Deng, Tolga Birdal, and Slobodan Ilic. PPFNet: Global context aware local features for robust 3d point matching. In *CVPR*, 2018. 1

[15] Huong Quynh Dinh, Anthony Yezzi, and Greg Turk. Texture transfer during shape transformation. *ACM Transactions on Graphics*, 24(2):289–310, Apr. 2005. 1

[16] Nicolas Donati, Etienne Corman, and Maks Ovsjanikov. Deep Orientation-Aware Functional Maps: Tackling Symmetry Issues in Shape Matching. *CVPR "12 : IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 11, 12

[17] Nicolas Donati, Abhishek Sharma, and Maks Ovsjanikov. Deep geometric functional maps: Robust feature learning for shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8592–8601, 2020. 1, 2, 3, 4, 6, 7, 12

[18] Marvin Eisenberger, Zorah Lähner, and Daniel Cremers. Smooth shells: Multi-scale shape registration with functional maps. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12262–12271, 2020. 1, 6

[19] Marvin Eisenberger, David Novotný, Gael Kerchenbaum, Patrick Labatut, Natalia Neverova, Daniel Cremers, and Andrea Vedaldi. Neuromorph: Unsupervised shape interpolation and correspondence in one go. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7469–7479, 2021. 2, 6, 12

[20] Marvin Eisenberger, Aysim Toker, Laura Leal-Taixé, and Daniel Cremers. Deep shells: Unsupervised shape correspondence with optimal transport. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 10491–10502. Curran Associates, Inc., 2020. 1, 2, 6, 7, 12

[21] Davide Eynard, Emanuele Rodola, Klaus Glashoff, and Michael M Bronstein. Coupled functional maps. In *3D Vision (3DV)*, pages 399–407. IEEE, 2016. 1, 2

[22] Danielle Ezuz and Mirela Ben-Chen. Deblurring and denoising of maps between shapes. In *Computer Graphics Forum*. Wiley Online Library, 2017. 4

[23] Matthias Fey, Jan Eric Lenssen, Frank Weichert, and Heinrich Müller. Splinecnn: Fast geometric deep learning with continuous b-spline kernels. In *Proceedings of the IEEE Conference*

on Computer Vision and Pattern Recognition, pages 869–877, 2018. 12

[24] Dvir Ginzburg and Dan Raviv. Cyclic functional mapping: Self-supervised correspondence between non-isometric deformable shapes, 2019. 2

[25] Dvir Ginzburg and Dan Raviv. Cyclic functional mapping: Self-supervised correspondence between non-isometric deformable shapes. In *Computer Vision – ECCV 2020*, pages 36–52. Springer International Publishing, 2020. 2

[26] Zan Gojcic, Caifa Zhou, Jan D. Wegner, and Andreas Wieser. The perfect match: 3D point cloud matching with smoothed densities. In *CVPR*, 2019. 1

[27] Shunwang Gong, Lei Chen, Michael Bronstein, and Stefanos Zafeiriou. Spiralnet++: A fast and highly efficient mesh convolution operator. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 3

[28] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 3d-coded: 3d correspondences by deep deformation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 230–246, 2018. 2, 11

[29] Yulan Guo, Mohammed Bennamoun, Ferdous Sohel, Min Lu, Jianwei Wan, and Ngai Ming Kwok. A comprehensive performance evaluation of 3d local feature descriptors. *International Journal of Computer Vision*, 116(1):66–89, 2016. 2

[30] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 2

[31] Oshri Halimi, Or Litany, Emanuele Rodola, Alex M Bronstein, and Ron Kimmel. Unsupervised learning of dense shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4370–4379, 2019. 1, 2, 12

[32] Qixing Huang, Fan Wang, and Leonidas Guibas. Functional map networks for analyzing and exploring large shape collections. *ACM Transactions on Graphics (TOG)*, 33(4):36, 2014. 1

[33] Vladimir G. Kim, Yaron Lipman, and Thomas Funkhouser. Blended intrinsic maps. *ACM Transactions on Graphics*, 30(4):1–12, July 2011. 6

[34] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 10

[35] Ilya Kostrikov, Zhongshi Jiang, Daniele Panozzo, Denis Zorin, and Joan Bruna. Surface networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2540–2548, 2018. 12

[36] Artiom Kovnatsky, Michael M Bronstein, Alexander M Bronstein, Klaus Glashoff, and Ron Kimmel. Coupled quasi-harmonic bases. In *Computer Graphics Forum*, volume 32, pages 439–448. Wiley Online Library, 2013. 1, 2

[37] Lei Li, Souhaib Attaiki, and Maks Ovsjanikov. SRFeat: Learning locally accurate and globally consistent non-rigid shape correspondence. In *2022 International Conference on 3D Vision (3DV)*. IEEE, Sept. 2022. 1, 2, 11, 12

[38] Or Litany, Tal Remez, Emanuele Rodola, Alex Bronstein, and Michael Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the IEEE international conference on computer vision*, pages 5659–5667, 2017. 1, 2, 4, 11

[39] O. Litany, E. Rodolà, A. M. Bronstein, and M. M. Bronstein. Fully spectral partial shape matching. *Computer Graphics Forum*, 36(2):247–258, May 2017. 1, 2

[40] O. Litany, E. Rodolà, A. M. Bronstein, M. M. Bronstein, and D. Cremers. Non-rigid puzzles. *Computer Graphics Forum*, 35(5):135–143, Aug. 2016. 1

[41] Roee Litman and Alexander M Bronstein. Learning spectral descriptors for deformable shape correspondence. *IEEE transactions on pattern analysis and machine intelligence*, 36(1):171–180, 2013. 1

[42] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework, 2022. 3

[43] Riccardo Marin, Souhaib Attaiki, Simone Melzi, Emanuele Rodolà, and Maks Ovsjanikov. Why you should learn functional basis, 2021. 1, 6

[44] Riccardo Marin, Marie-Julie Rakotosaona, Simone Melzi, and Maks Ovsjanikov. Correspondence learning via linearly-invariant embedding. *ArXiv*, abs/2010.13136, 2020. 1

[45] Haggai Maron, Meirav Galun, Noam Aigerman, Miri Trope, Nadav Dym, Ersin Yumer, Vladimir G Kim, and Yaron Lipman. Convolutional neural networks on surfaces via seamless toric covers. *ACM Trans. Graph.*, 36(4):71–1, 2017. 3, 6, 8

[46] Jonathan Masci, Davide Boscaini, Michael Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 37–45, 2015. 2, 3

[47] S. Melzi, R. Marin, E. Rodolà, U. Castellani, J. Ren, A. Poulenard, P. Wonka, and M. Ovsjanikov. Matching humans with different connectivity. *Eurographics Workshop on 3D Object Retrieval*, 2019. 6

[48] Simone Melzi, Jing Ren, Emanuele Rodolà, Abhishek Sharma, Peter Wonka, and Maks Ovsjanikov. ZoomOut: Spectral upsampling for efficient shape correspondence. *ACM Transactions on Graphics*, 38(6):1–14, Nov 2019. 1, 2, 3

[49] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Mathematics and Visualization*, pages 35–57. Springer Berlin Heidelberg, 2003. 3

[50] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5115–5124, 2017. 2

[51] Dorian Nogneng and Maks Ovsjanikov. Informative descriptor preservation via commutativity for shape matching. *Computer Graphics Forum*, 36(2):259–267, May 2017. 2

[52] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps. *ACM Transactions on Graphics*, 31(4):1–11, Aug. 2012. 2

[53] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional Maps: A Flexible Representation of Maps Between Shapes. *ACM Transactions on Graphics (TOG)*, 31(4):30, 2012. 1

[54] Maks Ovsjanikov, Etienne Corman, Michael Bronstein, Emanuele Rodolà, Mirela Ben-Chen, Leonidas Guibas, Frederic Chazal, and Alex Bronstein. Computing and processing correspondences with functional maps. In *ACM SIGGRAPH 2017 Courses*. ACM, July 2017. 2, 4

[55] Gautam Pai, Jing Ren, Simone Melzi, Peter Wonka, and Maks Ovsjanikov. Fast sinkhorn filters: Using matrix scaling for non-rigid shape correspondence with functional maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 384–393, June 2021. 2, 3, 4, 5

[56] Leonid Pishchulin, Stefanie Wuhrer, Thomas Helten, Christian Theobalt, and Bernt Schiele. Building statistical shape spaces for 3d human modeling. *Pattern Recognition*, 67:276–286, July 2017. 1

[57] Adrien Poulenard and Maks Ovsjanikov. Multi-directional geodesic neural networks via equivariant convolution. *ACM Transactions on Graphics (TOG)*, 37(6):1–14, 2018. 2, 3

[58] Adrien Poulenard, Marie-Julie Rakotosaona, Yann Ponty, and Maks Ovsjanikov. Effective rotation-invariant point cnn with spherical harmonics kernels. In *2019 International Conference on 3D Vision (3DV)*, pages 47–56. IEEE, 2019. 8, 12

[59] Adrien Poulenard, Primoz Skraba, and Maks Ovsjanikov. Topological Function Optimization for Continuous Shape Matching. *Computer Graphics Forum*, 37(5):13–25, 2018. 2

[60] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 1(2):4, 2017. 3

[61] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5105–5114, 2017. 3

[62] Jing Ren, Simone Melzi, Maks Ovsjanikov, and Peter Wonka. Maptree. *ACM Transactions on Graphics*, 39(6):1–17, Dec 2020. 2

[63] Jing Ren, Simone Melzi, Peter Wonka, and Maks Ovsjanikov. Discrete optimization for shape matching. *Computer Graphics Forum*, 40(5):81–96, Aug. 2021. 3

[64] Jing Ren, Simone Melzi, Peter Wonka, and Maks Ovsjanikov. Discrete optimization for shape matching. In *Computer Graphics Forum*. Wiley Online Library, 2021. 4

[65] Jing Ren, Adrien Poulenard, Peter Wonka, and Maks Ovsjanikov. Continuous and orientation-preserving correspondences via functional maps. *ACM Transactions on Graphics (ToG)*, 37(6):1–16, 2018. 1, 2, 6

[66] Jing Ren, Adrien Poulenard, Peter Wonka, and Maks Ovsjanikov. Continuous and orientation-preserving correspondences via functional maps. *ACM Transactions on Graphics (TOG)*, 37(6), 2018. 3

[67] E. Rodolà, L. Cosmo, M. M. Bronstein, A. Torsello, and D. Cremers. Partial functional correspondence. *Computer Graphics Forum*, 36(1):222–236, Feb. 2016. 1

[68] Emanuele Rodolà, Luca Cosmo, Michael M Bronstein, Andrea Torsello, and Daniel Cremers. Partial functional correspondence. In *Computer Graphics Forum*, volume 36, pages 222–236. Wiley Online Library, 2017. 2

[69] Jean-Michel Roufosse, Abhishek Sharma, and Maks Ovsjanikov. Unsupervised deep learning for structured shape matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1617–1627, 2019. 2, 3, 12

[70] Raif M Rustamov, Maks Ovsjanikov, Omri Azencot, Mirela Ben-Chen, Frédéric Chazal, and Leonidas Guibas. Map-based exploration of intrinsic shape differences and variability. *ACM Transactions on Graphics (TOG)*, 32(4):1–12, 2013. 4

[71] Samuele Salti, Federico Tombari, and Luigi Di Stefano. SHOT: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125:251–264, Aug. 2014. 1, 2

[72] Abhishek Sharma and Maks Ovsjanikov. Weakly supervised deep functional maps for shape matching. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19264–19275. Curran Associates, Inc., 2020. 1, 2, 6, 7, 12

[73] Nicholas Sharp, Souhaib Attaiki, Keenan Crane, and Maks Ovsjanikov. Diffusionnet: Discretization agnostic learning on surfaces. *ACM Trans. Graph.*, 01(1), 2022. 1, 2, 3, 4, 6, 10

[74] Nicholas Sharp and Keenan Crane. A laplacian for non-manifold triangle meshes. In *Computer Graphics Forum*, volume 39, pages 69–80. Wiley Online Library, 2020. 10

[75] Meged Shoham, Amir Vaxman, and Mirela Ben-Chen. Hierarchical functional maps between subdivision surfaces. *Computer Graphics Forum*, 38(5):55–73, Aug. 2019. 1

[76] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum*, pages 1383–1392. Wiley Online Library, 2009. 1, 2, 12

[77] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 3

[78] Giovanni Trappolini, Luca Cosmo, Luca Moschella, Riccardo Marin, Simone Melzi, and Emanuele Rodolà. Shape registration in the time of transformers. *Advances in Neural Information Processing Systems*, 34:5731–5744, 2021. 11

[79] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 2019. 2, 3, 6, 10

[80] Ruben Wiersma, Elmar Eisemann, and Klaus Hildebrandt. Cnns on surfaces using rotation-equivariant features. *ACM Transactions on Graphics (TOG)*, 39(4):92–1, 2020. 3, 11

[81] Ruben Wiersma, Ahmad Nasikun, Elmar Eisemann, and Klaus Hildebrandt. DeltaConv. *ACM Transactions on Graphics*, 41(4):1–10, July 2022. 2, 3, 6, 7, 10

[82] Yan Wu, Jun Yang, and Jinlong Zhao. Partial 3d shape functional correspondence via fully spectral eigenvalue alignment

and upsampling refinement. *Computers & Graphics*, 92:99–113, Nov. 2020. 1

[83] Rui Xiang, Rongjie Lai, and Hongkai Zhao. A dual iterative refinement method for non-rigid shape matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15930–15939, June 2021. 1, 3

[84] Zi Jian Yew and Gim Hee Lee. RPM-Net: Robust point matching using learned features. In *CVPR*, 2020. 1

[85] Andy Zeng, Shuran Song, Matthias Niessner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3DMatch: Learning local geometric descriptors from rgb-d reconstructions. In *CVPR*, 2017. 1

[86] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *Computer Vision – ECCV 2016*, pages 766–782. Springer International Publishing, 2016. 1

[87] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael J. Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 6