NCP: Neural Correspondence Prior for Effective Unsupervised Shape Matching

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

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

Abstract

We present Neural Correspondence Prior (NCP), a new paradigm for computing correspondences between 3D shapes. Our approach is fully unsupervised and can lead to high-quality correspondences even in challenging cases such as sparse point clouds or non-isometric meshes, where current methods fail. Our first key observation is that, in line with neural priors observed in other domains, recent network architectures on 3D data, *even without training*, tend to produce pointwise features that induce plausible maps between rigid or non-rigid shapes. Secondly, we show that given a noisy map as input, training a feature extraction network with the input map as supervision tends to remove artifacts from the input and can act as a powerful correspondence denoising mechanism, both between individual pairs and within a collection. With these observations in hand, we propose a two-stage unsupervised paradigm for shape matching, by (i) performing unsupervised training by adapting an existing approach to obtain an initial set of noisy matches, and (ii) using these matches to train a network in a supervised manner. We demonstrate that this approach significantly improves the accuracy of the maps, especially when trained within a collection. We show that NCP is data-efficient, fast, and achieves state-of-the-art results on many tasks. Our code can be found online: https://github.com/pvnieo/NCP.

1 Introduction

Establishing dense correspondences between 3D shapes is a fundamental problem in computer vision and computer graphics, as it enables many downstream applications such as statistical shape analysis [1, 2], texture [3] and deformation transfer [4, 5], and registration [6], to name a few.

A particularly challenging setting for this task is the computation of 3D correspondences between shapes that undergo non-rigid, non-isometric deformations, and that may exhibit some partiality, such as missing semantic parts and can be represented as sparse point clouds.

The standard approach is to formulate shape correspondence as a supervised learning problem, by relying on ground truth maps within large datasets of shape pairs. Given such ground truth, it is possible to train neural networks to either produce deformation fields [7], segmentation maps [8, 9, 10] or functional maps [11, 12, 13] between the input shapes. However, all these methods rely on the presence of labeled point-to-point correspondences, which are expensive to obtain, and are only available in a handful of cases.

At the same time, *unsupervised* techniques try to solve the matching problem by imposing structural properties on the maps, either in the intrinsic [14, 15, 16] or extrinsic domains [17, 18]. However, the priors used by these methods tend to be purely geometric (e.g., near isometry, divergence-free deformation fields), and, as we demonstrate below, are not always applicable, especially in challenging non-isometric settings.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

In this work, we demonstrate that the neural network itself can act as a powerful prior for shape correspondence problems. Specifically, we first observe that even without training, recent architectures for 3D shape analysis [19, 20, 21] produce pointwise features that capture local geometry and lead to well-structured maps between shapes. Secondly and perhaps more importantly, we demonstrate an effect that we call *neural correspondence prior*. Specifically, given noisy input maps, we formulate a supervised learning problem, where we aim to learn pointwise features that, when compared, would recover the input maps. Remarkably, we show that the correspondences computed by the trained networks are typically of higher quality than the input. Our results are consistent with the notion of *noise impedance* observed in other works on neural priors, pioneered in [22], and *neural bias* [23], which states that neural networks tend to optimize lower frequency signals before higher-frequency ones. We show that in the context of shape matching, these effects imply that neural networks can learn powerful features by training to overfit to given noisy or incoherent correspondences.

Based on these observations, we develop a two-stage algorithm for unsupervised shape matching. In the first stage, we adapt an existing unsupervised neural network, using variants of standard methods from the literature. We then train a new network from scratch in a supervised manner using the noisy maps as ground truth. We demonstrate that this second stage not only helps to denoise the maps but also improves the matching result overall and achieves state-of-the-art results on multiple tasks. We show that *neural correspondence prior* is applicable within both collections and on individual shape pairs, to provide test time refinement of dense input maps. Since this method makes no assumptions about the topology of the shapes, it can be used for complex cases such as non-isometric matching.

Overall, our contributions can be summarized as follows:

- We demonstrate that untrained neural networks for 3D shapes can produce pointwise features that are on par with complex axiomatic local descriptors.
- We show that when trained using artifact-laden maps for supervision, the features learned by neural networks lead to correspondences that are more coherent and of higher quality than the input. We call this effect *Neural Correspondence Prior*.
- Based on these insights, we develop a two-stage algorithm for unsupervised 3D non-rigid shape matching that achieves SOTA results for difficult matching scenarios such as non-isometric and point cloud matching.
- Using our unsupervised point-to-point correspondence method, we propose an approach for few-shot keypoint detection.

2 Related Work

Shape matching is a well-studied area in computer vision and graphics, and a full overview is beyond the scope of this paper. We refer the interested reader to recent surveys [24, 25, 26, 27] for a more in-depth treatment of this field. Below, we review methods that are most related to our work, with a focus on learning-based techniques, both supervised and unsupervised.

3D Non-Rigid Shape Correspondence There is a vast literature on learning non-rigid shape correspondences. In terms of supervised methods, an important direction is to consider the matching problem as a dense semantic segmentation problem where the labels are vertices on some template shape [8, 9, 10], or by mapping via template deformation [7]. However, these techniques are known to require a considerable amount of data, and may fail when discretization changes [19, 11]. Another approach is to use the functional map framework that was introduced in [28] and extended in many follow-up works [29, 30, 31, 32, 33, 34, 35, 36, 37] to name a few (see [38] for an overview). The functional map (fmap) framework is based on encoding and optimizing maps as small matrices in a reduced basis. The fmap formulation has been successfully used in the supervised setting to establish correspondences between complete and partial non-rigid shapes [11, 12, 13]. However, it is typically restricted to near-isometric matching between relatively clean discretizations.

The functional map framework has also been adapted to the unsupervised setting, either by imposing structural properties on the functional maps such as bijectivity, orthogonality, and commutativity with the Laplacian [15, 39], or by enforcing cycle consistency [40], geodesic distortion preservation [14], or by combining intrinsic and extrinsic properties [17, 18]. However, these methods all assume that the shapes are nearly isometric, not partial, and thus fail in these contexts. Another line of work

proposes to do unsupervised shape matching by reconstructing the input shapes in canonical order. These approaches are based on the encoder-decoder architecture, and find matches either by nearest neighbor in the latent space [41], or by nearest neighbor on the reconstruction of the two input shapes [42, 43]. However, these methods require a lot of shapes and learning time, and have only successfully been applied to man-made shapes that do not undergo significant non-rigid deformations.

Neural Prior The neural prior was first introduced in the seminal work of Ulyanov *et al.*[22], where the authors introduced the Deep Image Prior (DIP), and showed that a randomly initialized convolutional neural network can serve as a good prior for many 2D inverse problems, such as inpainting and denoising. Since then, several modifications and improvements have been made to DIP [44, 45, 46, 47, 48], in order to improve its performance or adapt it to new tasks. Another line of work [49, 50] has theoretically studied DIP either through the lens of regularization theory or through its connection to Gaussian processes. Despite its success in 2D computer vision, neural prior has been little exploited in the 3D domain, the only application being shape reconstruction. Indeed, [51, 52] attempt to reconstruct a water-tight mesh from a point cloud, either by overfitting several local patches to the point cloud or by shrinking an input mesh by a neural network. The objective of our work is to fill this gap and to propose a neural prior technique for 3D matching.

Unsupervised Keypoint Detection Unlike the 2D case, [53, 54, 55, 56], unsupervised 3D keypoint detection is relatively under-explored in the literature. The recently introduced KeyPointNet [57] benchmark provides a good testing bed. Prominent unsupervised keypoint detection methods are based on a self-supervised paradigm using an encoder-decoder network, with a reconstruction error. The encoder of Skeleton Merger [58] predicts a set of salient keypoints, that the decoder uses to produce a skeleton of the shape. UKPGAN [59] harnesses the power of GANs to detect keypoints, by forcing the decoder to reconstruct the input shape using solely the set of keypoints produced by the encoder. Finally, KeypointDeformer [60] predicts a set of keypoints that allow for efficient shape manipulation. In particular, given a source and a target shape, the keypoints predicted on the source shape are used to deform it to the target shape using a cage skinning deformer [61].

3 Motivation & Method Overview

Our main goal is to develop an unsupervised learning-based shape correspondence method that would take as input a pair of shapes $\mathcal{M}$ and $\mathcal{N}$ and produce a dense correspondence, also called a map, $T : \mathcal{M} \to \mathcal{N}$. For this, we explore the direction of neural priors and show how they can be used for 3D shape matching. The motivation behind this direction is twofold. First, as in the 2D case [62, 46], we hope that the structure of an untrained 3D neural network captures the low-level statistics of a single 3D shape, and thus can enable fine-tunning using a single pair. Second, as demonstrated in DIP [22], deep neural networks have high noise impedance, which allows them to learn good features from noisy inputs. This property can be used to guide the network to good local minima while trying to use, in our case, noisy point-to-point (p2p) matches for supervision.

Feature embeddings. In this work, we formulate the shape-matching problem for both man-made and organic shapes, by learning pointwise features. Such features can be used to compute correspondences either through nearest neighbor search or with minimal post-processing. We focus on features that are learned by a neural network. Specifically, the network N takes as input a 3D shape $\mathcal{M}$ and produces, as output, a feature vector $N(\mathcal{M})_x \in \mathbb{R}^d$, for every point of the shape $x \in \mathcal{M}$. When considering the entire object as a whole, we call the set $N(\mathcal{M})_x$ for all $x \in \mathcal{M}$ a *feature embedding* of shape $\mathcal{M}$.

Our method is based on two main observations: *neural bias* for untrained networks, and an effect that we call *neural correspondence prior*. Below we describe each of these effects and then describe our unsupervised shape-matching method in Sec. 4.1.

3.1 Neural bias for pointwise features

Our first observation is that feature embeddings produced by neural networks have a particular structure, which can be exploited in the context of computing features for shape correspondence. To highlight this effect we performed the following experiment. We take the test set of the FAUST-Remeshed (**FR**) and SCAPE-Remeshed (**SR**) datasets introduced in [33], and used in many recent works [11, 17, 39], and compute the correspondences using the pointwise features extracted by a DiffusionNet network [19] with randomly set weights. We use the default variant of DiffusionNet, which takes as input the XYZ coordinates of the shapes and produces a 128-dimensional feature

Table 1: Matching accuracy of DiffusionNet *with random weights*, compared to multiple shape matching methods, on the test set of **FR** and **SR**. Values are mean geodesic error $\times 100$ on unit-area shapes. We report the mean and standard deviation of three random runs.

Method / Dataset	Method Type	**FR**	**SR**
HKS [69] + NN	*Axiom*	26.8	28.6
WKS [70] + NN	*Axiom*	24.8	26.0
SHOT [71] + NN	*Axiom*	37.7	38.4
DiffusionNet [19] + NN	*Rand.init*	17.8 ± 0.1	20.3 ± 0.3
BCICP [33]	*Axiom*	15.	16.
FMNet [13]	*Sup*	11.	17.
SURFMNet [15]	*Unsup*	15.	12.
Unsup FMNet [14]	*Unsup*	10.	16.
HKS + FMAP	*Axiom*	20.2	26.9
WKS + FMAP	*Axiom*	20.1	27.1
SHOT + FMAP	*Axiom*	19.8	20.0
DiffusionNet + FMAP	*Rand.init*	$\mathbf{8.9 \pm 0.02}$	$\mathbf{13.9 \pm 0.3}$

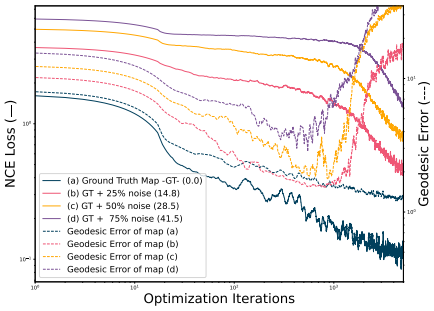

Figure 1: Learning curves showing the NCE loss for the ground truth map (GT), and some noisy maps with different levels of noise. Numbers in parentheses represent the geodesic error of the input maps. Observe the noise impedance and geodesic error decrease of intermediate maps.

vector for every point. We emphasize that the weights of the network are set randomly and *without any training*. We then used the extracted features to produce p2p maps either via a simple nearest neighbor (NN) search in the feature space or with the functional map framework [63] (FMAP). We compare the results to maps produced using the same procedure with classical axiomatic features such as SHOT [64], or using other axiomatic and training-based non-rigid correspondence methods.

As shown in Tab. 1, remarkably, the features computed with an *untrained* DiffusionNet network perform on par or better than axiomatic features and even outperform the *supervised* learning method, FMNet [65], based on training MLPs to refine pre-defined axiomatic features, as well as recent unsupervised extensions of that approach. We attribute the fact that a randomly-initialized DiffusionNet performs better than *trained* point-wise MLPs to the spatially-aware nature of the architecture of the network, which uses diffusion to simulate intrinsic convolution, thereby providing a strong neural prior [66, 67, 52]. Notice that using untrained features for matching was recently used in the context of 2D keypoint matching in [68].

3.2 NCP: Neural Correspondence Prior

While the previous effect relates to properties of untrained networks, in this work, we also observe another, complementary phenomenon that we call *Neural Correspondence Prior*, which we exploit in our approach below to train powerful feature extractors. This effect can be formulated as follows. Suppose we are given a fixed map $T_{\mathcal{MN}}$, between some shapes $\mathcal{M}$ and $\mathcal{N}$. We can formulate an optimization problem, where we aim to learn the feature embedding of $\mathcal{M}$ and $\mathcal{N}$ so that the induced map, e.g., computed via nearest neighbors in the feature space: i.e. $x \rightarrow \arg\min_y \|N(\mathcal{M})_x - N(\mathcal{N})_y\|$, is as close as possible to $T_{\mathcal{MN}}$. Note that unlike the neural bias mentioned in the previous section, here we formulate an optimization problem, where the parameters of the network are optimized to fit the given input map. Our key observation is that when the target map T is noisy or approximate, networks tend to produce well-structured feature embeddings until the very late stages of the optimization. I.e., throughout the early stages of optimization, the maps computed by the optimized features tend to be of *higher accuracy than the input* noisy map used for supervision.

To demonstrate the effect of NCP quantitatively, we performed an experiment in which we corrupted the ground truth map between a pair of shapes from the **FR** dataset, and then trained the DiffusionNet network, with a gradient descent optimizer using the standard NCE loss [72] to overfit to those corrupted maps. Specifically, to supervise this training, we use: (*a*) the ground truth (GT) map between the shapes, (*b*) the GT map where 25% of its entries have been assigned to random vertices, (*c*) the GT map with 50% noise, (*d*) the GT map with 75% noise. The results of this experiment are shown in Fig. 1. We plot both the NCE loss being optimized as well as the *geodesic error* of the intermediate maps with respect to the ground truth.

As shown in Fig. 1, while the network can easily adapt to the correct map, it has difficulty minimizing the loss in the case of noisy maps. Moreover, and perhaps more remarkably, observe that the geodesic

error starts to decrease *towards lower values than the input map*, in the first iterations of optimization and diverges only in the later stages, when the network starts to overfit to the noise in the map. This highlights that the intermediate maps tend to be of higher quality than the supervisory signal, and implies that such optimization with a neural network and early stopping can be used as an effective map denoising mechanism (Sec. 4.2). It should be mentioned that this effect does not depend on the shape pair, and other examples of different pairs are provided in Sec. 5.1.2 and the supplementary.

We observe this effect to hold very broadly in shape correspondence problems. For a single shape pair with a noisy input map, eventually, any map can be learned by the network if the parametrization and the number of training iterations are sufficient. However, the network architecture strongly regularizes the map search space, providing low impedance to 'signal' and high impedance to 'noise', resulting in an optimization trajectory that either converges to a good local minimum or passes near one [22]. Furthermore, in the presence of *shape collections*, this effect is even stronger and, as we demonstrate below, significantly helps to regularize computed features so that no early stopping or post-processing of the results becomes necessary (see our Algorithm 1 that we explain in detail in Sec. 4.1).

In addition to the network architecture itself, we attribute the *Neural Correspondence Prior* to the spectral bias principle [23, 73], which states that neural networks tend to learn low frequencies in the early stages of training and that low frequencies are more robust to random perturbations of network parameters. Since we formulate shape matching as the problem of computing optimal features, spectral bias, in our context, implies that even given a noisy or artifact-laden map as input, the *feature embeddings* produced by the optimized neural network tend *to be smooth*, especially in the early stages of the optimization. In other words, it is significantly harder for the network to produce a noisy, high-frequency feature embedding than a smooth one. Furthermore, a smooth feature embedding will tend to produce smooth correspondences between shapes, as suggested by the following theorem, which we state here and prove in the supplementary.

Theorem 1. *Let $\mathcal{M}$ and $\mathcal{N}$ be two compact smooth surfaces (smooth manifolds of dimension 2), and let $\mathbf{M}\colon \mathcal{M} \xrightarrow{\psi} \mathbb{R}^d$ (resp. $\mathbf{N}\colon \mathcal{N} \xrightarrow{\psi} \mathbb{R}^d$) be their respective feature embeddings produced by a neural network ψ. If the feature embeddings produced by ψ are injective and smooth, i.e., the mapping $\psi : \mathcal{M} \to \mathbb{R}^d, x \to \psi(x), \forall x \in \mathcal{M}$ is smooth and invertible on its image (and similarly for $\mathbf{N}$), **then** up to an arbitrarily small perturbation of $\mathcal{M}$ and $\mathcal{N}$, the map $T_{nn} : \mathcal{M} \to \mathcal{N}$ given by $T_{nn}(x) = \arg\min_{y \in \mathcal{N}} \|\psi(x) - \psi(y)\|$ is smooth up to sets of measure 0.*

This result highlights the fact that smooth feature embeddings, generically, translate into smooth maps, which is a desirable property in most typical correspondence scenarios.

4 Method Description

Based on the observations above, in this section, we introduce contexts in which the NCP can be exploited, resulting in two algorithms: one for unsupervised shape matching, and the second for p2p map denoising.

4.1 NCP within a shape collection

Given a collection of shapes $\{\mathcal{N}_i\}$, we propose to exploit the NCP in two ways. The first configuration assumes that we are also given a collection of artifact-laden maps $\{T_j\}$ between some shape pairs in $\{\mathcal{N}_i\}$ as input. In that case, we propose to train a randomly-initialized neural network to produce feature embedding for each shape, s.t., the induced maps are as close as possible to the target maps $\{T_j\}$. We provide the exact choice of the loss function depending on the nature of the underlying shapes in Sec. 5 below.

In the second configuration, no maps are given as input. In this case, we propose a two-stage pipeline where in the first stage, we adapt an existing unsupervised correspondence method to obtain possibly approximate very noisy initial correspondences. We then formulate a supervised learning problem, where we learn feature embeddings that would induce these computed correspondences as a target. Once the network is trained, we establish final correspondences via a simple nearest neighbor search in the feature space. Note that the trained network can be used to establish correspondences both between shapes within the given shape collection, but also *across new, never seen shape pairs.* In our evaluation below, and unless otherwise noted, we always evaluate on a test set, never seen during

Algorithm 1: NCP-UN: **N**eural-**C**orrespondence-**P**rior based **UN**supervised Shape Matching

1 **Input:** Collection of shapes $\{\mathcal{N}_i\}$
2 **Output:** p2p maps between shapes of the test set of $\{\mathcal{N}_i\}$

Stage 1
1: Train a variant of an off-the-shelf unsupervised matching method **UM** on the train set of $\{\mathcal{N}_i\}$
2: Predict p2p maps $\{T_i\}$ on the train set of $\{\mathcal{N}_i\}$ using trained **UM**

Stage 2
3: Use $\{T_i\}$ to supervise the training of a randomly-initialized neural network **RN** with a p2p loss
4: Predict Feature Embedding $\{\mathbf{RN}(\mathcal{N}_i)\}$ on the test set of $\{\mathcal{N}_i\}$ using the trained **RN**
5: Use $\{\mathbf{RN}(\mathcal{N}_i)\}$ to establish correspondences either using a nearest neighbor or functional map pipeline.

training. We dubbed the resulting algorithm **NCP-UN**, and we summarize it in Algorithm 1. This is the algorithm that is used throughout most of our experiments.

We argue that it is advantageous to perform the second step for the entire shape collection, instead of doing it for each pair individually for two main reasons: first, it is faster and avoids the need for test-time optimization (see Sec. 4.2), and second, the collection provides an additional layer of regularization. In fact, since we optimize for features on each shape, it is difficult to compute features that would overfit to all $O(N^2)$ noisy maps in a collection of N shapes, as the artifacts across pairs are typically not consistent. We observe that the noise impedance property is even stronger in the case of a collection, and the produced maps are better than if the refinement is done per pair.

Intuition behind Stage 2 of Algorithm 1 Our intuition for Stage 2 is threefold. First, and motivated by the results of Sec. 3.2 (the NCP effect), we showed that neural networks tend to have difficulty overfitting to corrupted inputs, and tend to favor smooth outputs. Thus, in the process of over-fitting to approximate maps by a neural network, the output maps tend to be of **better quality** than the input. This motivates the use of artifact-laden maps as supervision for a network thus capitalizing on the NCP effect to recover better correspondences. Second, as we mentioned above, given a collection of approximate maps, their errors tend to be inconsistent, and training a network using such maps for supervision adds an extra layer of regularization, as it is difficult for the network to adapt to all inconsistent errors on all pairs at once. Finally, in our formulation of Stage 2, we learn *one feature embedding per shape* and compute correspondences via nearest neighbor search between feature embeddings. This also adds strong regularization, since it makes it difficult for the network to learn a feature embedding, which can reproduce errors in the maps between all shape pairs.

For our applications (see Sec. 5), we used the following design choices. For the **UM** method, we used the unsupervised functional map method [15, 39], with a DiffusionNet [19] backbone for 3D meshes, and Point-MLP [74] backbone for point clouds. The same backbone was used as **RN** for step 3. in our Algorithm 1. Concerning the p2p loss in step 3, we used the LIE loss [75] for 3D meshes and point clouds. We also used a variant of our approach, dubbed NCP-UN$_{\text{fmap}}$, that uses the FMAP framework and loss from [11] in steps 3 and 5 of Algorithm 1, especially on organic shapes, where the Laplacian basis tends to be of high quality.

We want to emphasize, first, that in all of our applications below, we followed the best practices depending on the nature of shapes (e.g., organic shapes vs. point clouds), and second, our observations regarding NCP are independent of the choice of loss and network architecture. A demonstration of the general nature of NCP, as well as more details about the implementation, are provided in the supplementary. Across *all* cases, we observe a significant improvement in results when using the second stage of our NCP-UN pipeline (supervised learning) compared to the initial input maps.

Finally, we want to emphasize that it is possible to repeat Stage 2 several times, and we observed that there is a slight improvement in some cases, but most often it stagnates after the first iteration. We experimented with this and chose to keep the method simple and use only one iteration thus, avoiding an additional tunable hyperparameter.

4.2 Test Time Denoising

In addition to the previous setting, we also observe that NCP can be used to denoise a correspondence when given a single shape pair. Specifically, given a noisy initial map, we train a neural network to produce feature embeddings that would induce the given input map, used as supervision. Again, we tailor the exact choice of loss to the nature of the shapes and describe one option in Sec. 5.1.2 below.

Test time denoising can be advantageous in the absence of a shape collection. At the same time, as mentioned in the original DIP work [22], in the case of a single shape pair, early stopping must be used in order to stop training at the best local minimum. For the shape-matching task, we propose to use a cycle loss based on bijectivity as the criterion for stopping. Given a pair of shapes $\mathcal{M}, \mathcal{N}$, and two maps between them $\Pi_{\mathcal{MN}}$ and $\Pi_{\mathcal{NM}}$ computed via nearest neighbors between optimized features, and represented as soft binary matrices (see Eq. (4) in the supplementary materials for computation details), we compute $L_{cycle} = \|X_{\mathcal{N}} - \Pi_{\mathcal{NM}}\Pi_{\mathcal{MN}}X_{\mathcal{N}}\|_F^2$, where $X_{\mathcal{N}}$ is the matrix of XYZ coordinates of shape $\mathcal{N}$. We overfit the network for a number of iterations and stop when L_{cycle} stops improving after a predetermined patience period. The weights that produce the minimum value of L_{cycle} are used to compute the pointwise features that we then use for extracting the p2p map, using the nearest neighbors in the feature space.

4.3 Correspondences as a Tool for Downstream Tasks

Given computed correspondences, we also use them to solve multiple downstream tasks. Indeed, since our proposed algorithm is unsupervised, we can use its output, i.e., p2p maps, to transfer multiple quantities between shapes, such as keypoints. For this, we only need a few labeled examples with the quantities we want to transfer, resulting in a few-shot algorithm.

In the following, we present a method for few-shot keypoint detection, dubbed **FSKD** hereafter, and based on the p2p maps obtained by our unsupervised algorithm **NCP-UN**. A particular challenge in this setup is that some keypoints and parts appear exclusively on some shapes and not in others, and so to account for the keypoint appearing there, multiple labeled source shapes must be used.

We begin our **FSKD** by randomly selecting N shapes that cover the largest set of keypoints. These N shapes will be labeled and we transfer their keypoints to the shapes in the test set, in order to "discover" their keypoints.

Given a target shape, **FSKD** is composed of three steps: **1.** Detection of potential keypoints by transferring keypoints from labeled shapes, **2.** Filtering to remove keypoints that are likely not to exist on the target shape, **3.** Combination: merge transferred keypoints if multiple points on the target shape are assigned to the same keypoint ID. We provide the implementation details for these three steps in the supplementary (see Appendix C).

5 Results & Applications

In this section, we provide results in a wide range of challenging tasks, showing the efficiency and robustness of our approach to different types of data and tasks. In particular, we consider the tasks of shape correspondence between man-made shapes in Sec. 5.1.1, non-rigid shape correspondence in Sec. 5.1.2, part segmentation in Sec. 5.2, and few-shot keypoint detection in Sec. 5.3. The fact that our model does not rely on a geometric prior allows it to provide good results on different types of data, including isometric and non-isometric, organic, and man-made data.

Datasets We conduct our experiments on four different datasets, both on point clouds and 3D meshes. To the best of our knowledge, there is no point cloud dataset that provides dense point-to-point ground-truth correspondences. For this purpose, we follow the setup of [41], and use KeyPointNet dataset [57] for the man-made shape correspondence experiment in Sec. 5.1.1. We also use this dataset for our few-shot keypoint detection experiment in Sec. 5.3. KeypointNet is composed of over 8K models and 100K semantic keypoints labeled by professional humans. We use the same train/validation/test splits as the original paper [57]. For the part segmentation experiment in Sec. 5.2, we use the PartNet dataset [76], which is composed of 16881 shapes of 16 categories, where each category has a different number of parts ranging from 2 to 6.

For the 3D meshes, we use the organic, non-isometric non-rigid SMAL-based [77, 36] dataset and the SHREC'20 Non-Isometric benchmark [78] for our non-rigid correspondence experiment in Sec. 5.1.2, as these are challenging *non-isometric* datasets. The former dataset is composed of 50 animal shapes represented as 3D meshes. 25 shapes are used for training, and the remaining 25 are used for testing. It should be noted that the set of animals and poses seen in training is different from the one used in testing. The SHREC'20 benchmark consists of 14 scans of different animals, some of which contain holes and topological deformations. This, in addition to the fact that the ground truth maps are given as a set of sparsely annotated keypoints, means that most supervised methods cannot be applied.

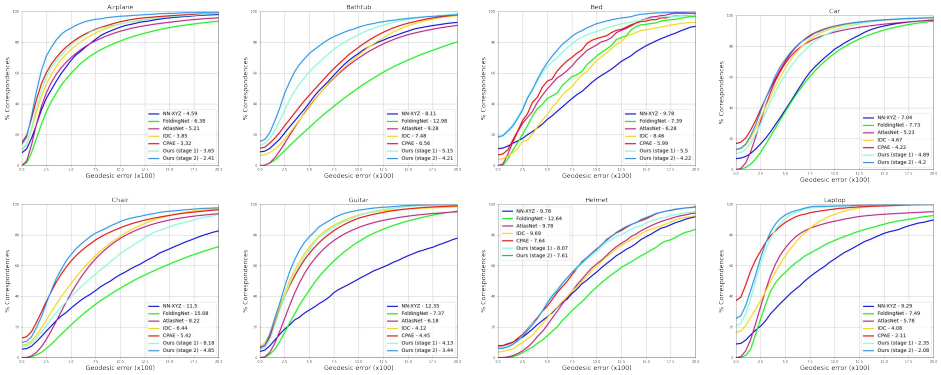

Figure 2: Correspondence accuracy on 8 categories in the KeypointNet dataset.

5.1 3D Shape Correspondences

5.1.1 Correspondences on Point Clouds

We test the quality of the matches produced by NCP-UN on sparse point clouds using the KeyPointNet dataset. We train the first stage of our method using the unsupervised functional map method [15, 39], with a PointMLP backbone [79] and XYZ coordinates as input, and impose the bijectivity loss on the functional map. The second stage is trained using LIE loss, and maps are extracted using the nearest neighbor in the feature space. For testing, we follow the same setup as [41], by generating pairs of shapes between all point clouds in a given category and removing the pairs that do not share the same set of semantic keypoints. We compute the L2 distance between the transferred keypoint and the ground truth one. Our method is evaluated against state-of-the-art learning-based 3D dense correspondence prediction approaches, including AtlasNetV2 [80], FoldingNet [81], IDC [43], CPAE [41], in addition to the nearest neighbor baseline between the XYZ coordinates of the points, which turns out to be a competitive baseline, not considered by previous methods.

In Fig. 2, we report the percentage of testing pairs where the distances between predicted and ground truth maps are below a given threshold for 8 categories and report the remaining categories in the supplementary. It can be seen that our method outperforms all methods in the 8 categories, and achieves state-of-the-art results. Note that our method obtains SOTA results although the method used in stage 1 is not fully designed to work on point clouds (the quality of the Laplacian is not good, and the used bijectivity loss is weak). We believe that using a more adapted method for the first stage will further improve the results.

5.1.2 Correspondences Between Non-Rigid Meshes

Evaluation Protocol We follow the Princeton protocol [82] for evaluating all correspondences, as used in all recent works. Specifically, we compute the pointwise geodesic error between the predicted maps and the ground truth, normalized by the square root of the surface area of the target shape.

For the non-rigid non-isometric application, we trained the unsupervised functional map network from [19] and used LIE loss for the second stage (step 3. in Algorithm 1). P2P maps were obtained using nearest neighbors between feature embeddings. Since the SHREC'20 dataset is small (only 14 shapes), we follow previous work [83] by training and testing on the same set, whereas for SMAL, the training and testing sets are different. For a fair evaluation, we compared our method to many state-of-the-art supervised and unsupervised methods. For the supervised methods, we compared to FMNet [13], GeomFMaps [11], GeomFMaps + DiffusionNet [19] and LIE [75]. For the unsupervised methods, we compared our method to WSupFMNet [39], Smooth Shells [18], Deep Shells [17], NeuroMorph [83] and WSupFMNet + DiffusionNet [19].

As shown in Tab. 2 and Fig. 3, our method achieves state-of-the-art results among unsupervised methods and supervised methods. The majority of unsupervised methods and supervised methods fail because of the near-isometry assumption they make. Since our method makes no assumptions about the topology of the shape, it can perform very well in difficult scenarios such as non-isometric animals. It should be noted that these results would not be possible without the second stage of our algorithm, which is enabled by NCP. For example, for the SMAL dataset, the result of the first stage

Table 2: Matching accuracy on the SMAL test set. Values are mean geodesic error $\times 100$ on unit-area shapes

	Method	**SMAL**
Unsup	WSupFMNet [39]	13.3
	Smooth Shells [18]	16.3
	DeepShells [17]	15.2
	NeuroMorph [83]	23.1
	Ours (Stage1)	7.8
Sup	FMNet [13]	47.1
	GeomFMaps [11]	8.8
	GeomFMaps + DiffusionNet [19]	6.6
	LIE [75]	20.0
Unsup	Ours (Stage2)	6.0
	Ours (Stage2 + fmap)	**5.8**
	Ours (Test-Time Optimization)	6.4

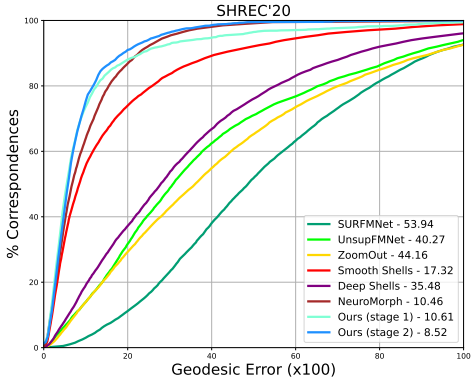

Figure 3: Correspondence quality of different methods on the test set of SHREC'20 Non-Isometric dataset.

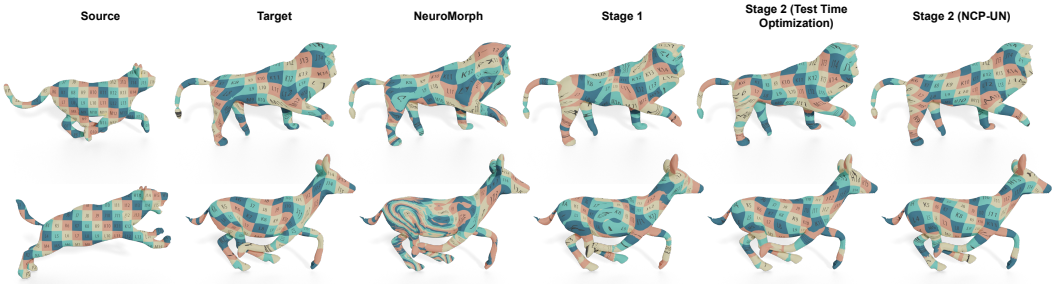

Figure 4: Qualitative results on SMAL dataset, showing the quality of the maps produced by Stage 1, and refined by Stage 2, either using Test Time Optimization (see Sec. 4.2), or using NCP-UN (see Algorithm 1).

is **7.8**, and the second stage improves this result to **5.8**, which is a **25%** improvement, achieving state-of-the-art results.

We provide in Fig. 4 some qualitative results showing the performance of our algorithm against the state-of-the-art NeuroMorph method on the SMAL dataset. It can be seen that our method produces visually plausible maps, especially after applying the second stage.

Test Time Optimization To show the utility of test time optimization, as well as the early stopping criteria we introduced, the performed the following experiment. After training the first stage of our algorithm, the trained model was used to predict p2p on the test set of **SMAL** dataset. Pairs of shapes are then created, and we use our test time optimization method (see Sec. 4.2) to improve upon the maps produced by the first stage. Results are reported in Tab. 2, see row "Ours (Test-Time Optimization)". It can be seen that similarly to the previous case, training a random network on top of artifact-laden maps helps to denoise them. We also show in Fig. 4 some qualitative results that illustrate the effect of test time optimization and compare it to NCP-UN.

The results of this experiment confirm our hypothesis that doing the second stage on a collection, instead of individual pairs, provides a second layer of regularization, as the results are better, in addition to improving computational complexity. In fact, the second stage using the whole collection takes around **15 minutes**, meanwhile, doing it on each pair individually takes around **160 minutes**, which represents more than $10\times$ improvement in running time.

5.2 Part Segmentation Transfer

We further validate our approach on the part label transfer task on the PartNet dataset following the setup of [41]. Result are summarized in Tab. 3, where the average intersection-over-union IOU is reported. Our method outperforms the baselines on 14 out of 16 categories, achieving state-of-the-art result by more than 3% IOU on average. In classes with large intra-class variations such as Lamps, our method outperforms the baselines by up to 8 % average IOU points. In addition to being performant,

Table 3: Part label transfer (in the average IOU(%)) for 16 categories in the ShapeNet part dataset.

	pla.	bag	cap	car	cha.	ear.	gui.	kni.	lam.	lap.	bik.	mug.	pis.	roc.	ska.	tab.	avg.
IDC [43]	60.1	56.2	59.7	-	72.2	45.3	**81.5**	66.4	42.6	88.5	40.5	87.5	**66.4**	37.2	50.7	70.4	61.7
CPAE [41]	61.3	59.3	61.6	-	72.6	55.5	78.9	71.3	53.2	89.9	55.4	86.5	66.2	40.2	61.8	72.5	65.8
Ours	**63.7**	**66.7**	**68.7**	**57.4**	**80.2**	**59**	78.8	**72.5**	**61.9**	**91.4**	**57.2**	**89.5**	61.4	**44.2**	**63.6**	**79.2**	**69.2**

Figure 5: **(a)** mIoU results of our simple few-shot keypoint detection algorithm on KeypointNet using only 3 labeled examples. Our simple method achieves good results and is on par with or better than recent keypoint detection methods. **(b)** Qualitative results on the table category, showing the shapes used for training (N=3, left), human annotations on the testset (center) as well as our predictions (right) on four examples.

our method is extremely fast. In fact, training both stages of our method takes on average **1 hour** (see the the computational specifications in Appendix B), which is more efficient in comparison to the baselines, where CPAE for example takes around 24 hours to be trained for one category.

5.3 Few-Shot 3D KeyPoint Detection

We evaluated our few-shot keypoint detection algorithm **FSKD** on the KeyPointNet dataset, following the setup of [59], and using only *3 labeled source shapes* (the resulting algorithm is thus a three-shot keypoint detection algorithm). The goal of this task is to predict keypoints that correlate with human annotation. The performance is evaluated by the mean Intersection over Union (mIoU). The intersection is counted if the geodesic distance between a predicted keypoint and its nearest ground truth keypoint is less than some geodesic threshold. The union is simply the union of the detected keypoints and ground truth keypoints.

We compare **FSKD** against multiple baselines, including SIFT-3D [84], HARRIS-3D [85], ISS [86], D3Feat [87], USIP [88] and UKPGAN [59]. Results are summarized in Fig. 5 - left. It can be seen that our simple, correspondence-based method outperforms the baselines tailored specifically for this task. In particular, our method achieves state-of-the-art results for a threshold superior to 0.035, using only **three** labeled shapes. We provide some qualitative results in Fig. 5 - right.

6 Conclusion & Limitations

In this work, we showed that using noisy maps as a supervisory neural network training signal for the 3D shape-matching task, can lead to significantly higher quality correspondences. We named this effect Neural Correspondence Prior (NCP). Through extensive experiments, we shed light on the properties of NCP and developed a two-stage algorithm for unsupervised 3D shape matching. We demonstrate the effectiveness and generality of our algorithm on a range of shape-matching problems, including unsupervised shape matching on both man-made and organic non-rigid shapes, achieving state-of-the-art results on a wide range of benchmarks.

One limitation of our approach is that it assumes that the p2p maps used to train the second stage of the algorithm still have some structure. In fact, if the unsupervised method used in the first stage fails completely and provides random maps, our method will not converge. This problem can be mitigated by using a good application-specific unsupervised method for the first stage, especially with the recent interest and increase in unsupervised 3D learning methods.

Acknowledgements We gratefully acknowledge the anonymous reviewers for their valuable suggestions. Parts of this work were supported by the ERC Starting Grant No. 758800 (EXPROTEA) and the ANR AI Chair AIGRETTE.

References

[1] 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

[2] 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

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

[4] Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. *ACM Transactions on Graphics*, 23(3):399–405, August 2004. 1

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

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

[7] 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. 1, 2

[8] 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. 1, 2

[9] 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. 1, 2

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

[11] 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, 6, 8, 9, 18, 22, 24

[12] Souhaib Attaiki, Gautam Pai, and Maks Ovsjanikov. DPFM: Deep partial functional maps. In *2021 International Conference on 3D Vision (3DV)*. IEEE, December 2021. 1, 2, 18

[13] 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, 8, 9, 22

[14] 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, 4

[15] 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. 1, 2, 4, 6, 8, 18

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

[17] 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, 3, 8, 9, 23

[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, 2, 8, 9, 23

[19] Nicholas Sharp, Souhaib Attaiki, Keenan Crane, and Maks Ovsjanikov. Diffusionnet: Discretization agnostic learning on surfaces. *ACM Trans. Graph.*, 01(1), 2022. 2, 3, 4, 6, 8, 9, 18, 19, 20, 23, 24

[20] 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, 23

[21] 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. 2, 22

[22] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. *arXiv:1711.10925*, 2017. 2, 3, 5, 7

[23] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 5301–5310. PMLR, 2019. 2, 5, 21

[24] Yusuf Sahillioglu. Recent advances in shape correspondence. *The Visual Computer*, 36:1705 – 1721, 2019. 2

[25] S. Biasotti, A. Cerri, A. Bronstein, and M. Bronstein. Recent trends, applications, and perspectives in 3d shape similarity assessment. *Computer Graphics Forum*, 35(6):87–119, October 2015. 2

[26] Bailin Deng, Yuxin Yao, Roberto M Dyke, and Juyong Zhang. A survey of non-rigid 3d registration. *arXiv preprint arXiv:2203.07858*, 2022. 2

[27] G. K. L. Tam, Zhi-Quan Cheng, Yu-Kun Lai, F. C. Langbein, Yonghuai Liu, D. Marshall, R. R. Martin, Xian-Fang Sun, and P. L. Rosin. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. *IEEE Transactions on Visualization and Computer Graphics*, 19(7):1199–1217, July 2013. 2

[28] 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. 2

[29] 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

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

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

[32] Meged Shoham, Amir Vaxman, and Mirela Ben-Chen. Hierarchical functional maps between subdivision surfaces. *Computer Graphics Forum*, 38(5):55–73, August 2019. 2

[33] 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. 2, 3, 4, 19, 21, 22, 24

[34] 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. 2

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

[36] Riccardo Marin, Souhaib Attaiki, Simone Melzi, Emanuele Rodolà, and Maks Ovsjanikov. Why you should learn functional basis, 2021. 2, 7, 20

[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, September 2022. 2

[38] 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

[39] 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. 2, 3, 6, 8, 9, 18

[40] 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

[41] An-Chieh Cheng, Xueting Li, Min Sun, Ming-Hsuan Yang, and Sifei Liu. Learning 3d dense correspondence via canonical point autoencoder, 2021. 3, 7, 8, 9, 10

[42] Zerong Zheng, Tao Yu, Qionghai Dai, and Yebin Liu. Deep implicit templates for 3d shape representation, 2020. 3

[43] Feng Liu and Xiaoming Liu. Learning implicit functions for topology-varying dense 3d shape correspondence, 2020. 3, 8, 10

[44] Reinhard Heckel and Paul Hand. Deep decoder: Concise image representations from untrained non-convolutional networks. *International Conference on Learning Representations*, 2019. 3

[45] Reinhard Heckel and Mahdi Soltanolkotabi. Compressive sensing with un-trained neural networks: Gradient descent finds the smoothest approximation. *ArXiv*, abs/2005.03991, 2020. 3

[46] Gary Mataev, Michael Elad, and Peyman Milanfar. Deepred: Deep image prior powered by red. *ArXiv*, abs/1903.10176, 2019. 3

[47] Jiaming Liu, Yu Sun, Xiaojian Xu, and Ulugbek S. Kamilov. Image restoration using total variation regularized deep image prior. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, May 2019. 3

[48] Daniel Otero Baguer, Johannes Leuschner, and Maximilian Schmidt. Computed tomography reconstruction using deep image prior and learned reconstruction methods. *Inverse Problems*, 36(9):094004, sep 2020. 3

[49] Sören Dittmer, Tobias Kluth, Peter Maass, and Daniel Otero Baguer. Regularization by architecture: A deep prior approach for inverse problems. *Journal of Mathematical Imaging and Vision*, 62(3):456–470, oct 2019. 3

[50] Zezhou Cheng, Matheus Gadelha, Subhransu Maji, and Daniel Sheldon. A bayesian perspective on the deep image prior, 2019. 3

[51] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3

[52] Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. Point2mesh. *ACM Transactions on Graphics*, 39(4), August 2020. 3, 4

[53] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Unsupervised learning of object landmarks through conditional image generation. In *NeurIPS*, 2018. 3

[54] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Self-supervised learning of interpretable keypoints from unlabelled videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[55] James Thewlis, Hakan Bilen, and Andrea Vedaldi. Unsupervised learning of object landmarks by factorized spatial embeddings. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 3

[56] Olivia Wiles, A. Sophia Koepke, and Andrew Zisserman. Self-supervised learning of a facial attribute embedding from video. In *BMVC*, 2018. 3

[57] Yang You, Yujing Lou, Chengkun Li, Zhoujun Cheng, Liangwei Li, Lizhuang Ma, Cewu Lu, and Weiming Wang. Keypointnet: A large-scale 3d keypoint dataset aggregated from numerous human annotations. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2020. 3, 7, 19, 22, 23

[58] Ruoxi Shi, Zhengrong Xue, Yang You, and Cewu Lu. Skeleton merger: an unsupervised aligned keypoint detector. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2021. 3

[59] Yang You, Wenhai Liu, Yanjie Ze, Yong-Lu Li, Weiming Wang, and Cewu Lu. Ukpgan: A general self-supervised keypoint detector, 2020. 3, 10

[60] Tomas Jakab, Richard Tucker, Ameesh Makadia, Jiajun Wu, Noah Snavely, and Angjoo Kanazawa. Keypointdeformer: Unsupervised 3d keypoint discovery for shape control. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2021. 3

[61] Wang Yifan, Noam Aigerman, Vladimir G Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. Neural cages for detail-preserving 3d deformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 75–83, 2020. 3

[62] Yeonsik Jo, Se Young Chun, and Jonghyun Choi. Rethinking deep image prior for denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5087–5096, October 2021. 3

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

[64] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In *European conference on computer vision*, pages 356–369. Springer, 2010. 4, 19

[65] 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. 4

[66] Adam Gaier and David Ha. Weight agnostic neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 4

[67] Andrew M. Saxe, Pang Wei Koh, Zhenghao Chen, Maneesh Bhand, Bipin Suresh, and Andrew Y. Ng. On random weights and unsupervised feature learning. In *In ICML*, 2011. 4

[68] Sunghwan Hong and Seungryong Kim. Deep matching prior: Test-time optimization for dense correspondence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9907–9917, October 2021. 4

[69] 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. 4, 19

[70] 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. 4, 19

[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, August 2014. 4

[72] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. PointContrast: Unsupervised pre-training for 3d point cloud understanding. In *European Conference on Computer Vision*, pages 574–591. Springer, 2020. 4, 19

[73] Yuan Cao, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu. Towards understanding the spectral bias of deep learning. In *IJCAI*, 2021. 5

[74] 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. 6, 19

[75] Riccardo Marin, Marie-Julie Rakotosaona, Simone Melzi, and Maks Ovsjanikov. Correspondence learning via linearly-invariant embedding. *ArXiv*, abs/2010.13136, 2020. 6, 8, 9, 19

[76] Li Yi, Vladimir G. Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *SIGGRAPH Asia*, 2016. 7, 19

[77] 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. 7, 20

[78] Roberto M. Dyke, Yu-Kun Lai, Paul L. Rosin, Stefano Zappalà, Seana Dykes, Daoliang Guo, Kun Li, Riccardo Marin, Simone Melzi, and Jingyu Yang. SHREC'20: Shape correspondence with non-isometric deformations. *Computers & Graphics*, 92:28–43, November 2020. 7, 20

[79] 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. In *International Conference on Learning Representations*, 2022. 8

[80] Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. Learning elementary structures for 3d shape generation and matching. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 8

[81] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. FoldingNet: Point cloud auto-encoder via deep grid deformation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, June 2018. 8

[82] Vladimir G. Kim, Yaron Lipman, and Thomas Funkhouser. Blended intrinsic maps. *ACM Transactions on Graphics*, 30(4):1–12, July 2011. 8

[83] 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. 8, 9, 23

[84] Blaine Rister, Mark A. Horowitz, and Daniel L. Rubin. Volumetric image registration from invariant keypoints. *IEEE Transactions on Image Processing*, 26(10):4900–4910, October 2017. 10

[85] Ivan Sipiran and Benjamin Bustos. Harris 3d: a robust extension of the harris operator for interest point detection on 3d meshes. *The Visual Computer*, 27(11):963–976, July 2011. 10

[86] Yu Zhong. Intrinsic shape signatures: A shape descriptor for 3d object recognition. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*. IEEE, September 2009. 10

[87] Xuyang Bai, Zixin Luo, Lei Zhou, Hongbo Fu, Long Quan, and Chiew-Lan Tai. D3Feat: Joint learning of dense detection and description of 3d local features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6359–6367, 2020. 10

[88] Jiaxin Li and Gim Hee Lee. USIP: Unsupervised stable interest point detection from 3d point clouds. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, October 2019. 10

[89] Nicholas Sharp and Keenan Crane. A laplacian for nonmanifold triangle meshes. In *Computer Graphics Forum*, volume 39, pages 69–80. Wiley Online Library, 2020. 18, 19

[90] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011. 19

[91] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 19

[92] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 20

[93] Mohammad Ghomi and Joel Spruck. Total curvature and the isoperimetric inequality in Cartan–Hadamard manifolds. *The Journal of Geometric Analysis*, 32(2):50, 2022. 21

[94] Robert L Foote. Regularity of the distance function. *Proceedings of the American Mathematical Society*, 92(1):153–155, 1984. 21

[95] Mantegazza and Mennucci. Hamilton—Jacobi equations and distance functions on Riemannian manifolds. *Applied Mathematics & Optimization*, 47(1):1–25, 2002. 21

[96] Frédéric Chazal and Rémi Soufflet. Stability and finiteness properties of medial axis and skeleton. *Journal of Dynamical and Control Systems*, 10(2):149–170, 2004. 21

[97] Victor Guillemin and Alan Pollack. *Differential topology*, volume 370. American Mathematical Soc., 2010. 21

[98] Charlotte Greenblatt. An introduction to transversality. `https://schapos.people.uic.edu/MATH549_Fall2015_files/Survey%20Charlotte.pdf`, 2015. 21

[99] Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Joern-Henrik Jacobsen. Invertible residual networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 573–582. PMLR, 09–15 Jun 2019. 21

[100] Ulrich Pinkall and Konrad Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, January 1993. 21, 23

[101] 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. 25

[102] James Klatzow, Giovanni Dalmasso, Neus Martínez-Abadías, James Sharpe, and Virginie Uhlmann. μmatch: 3d shape correspondence for biological image data. *Frontiers in Computer Science*, 4, February 2022. 25

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Sec. 6
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See supplementary
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See supplementary
   (b) Did you include complete proofs of all theoretical results? [Yes] See supplementary
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Our code can be found online: `https://github.com/pvnieo/NCP`
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Sec. 5 and Supp Mat
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] In some experiments when it's not prohibitively expensive
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Supplementary
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] The used datasets are publicly available.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Supplementary Materials for: NCP: Neural Correspondence Prior for Effective Unsupervised Shape Matching

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

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

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

Specifically, we first give a background on functional maps in Appendix A. Next, the implementation details are provided in Appendix B. The implementation details of our few-shot keypoint detection algorithm are presented in Appendix C. In Appendix D, we provide a complete formulation and proof of the theorem we introduced in Sec. 3.2 of the main text. In Appendix E, we provide an experiment to verify the conditions of the aforementioned theorem. We present a more in-depth analysis of the Neural Correspondence Prior (NCP) effect in Appendix F. Additional quantitative and qualitative results for the shape matching on man-made data tasks are included in Appendix G. An experiment showing the performance of our method in the case of near isometric data is presented in Appendix H Finally, we discuss the societal impact of our work in Appendix I.

A Background On Functional Maps & Notation

Our work uses the functional map framework as a first estimator for p2p maps, and multiple losses on point-to-point (p2p) maps to learn robust features that allow extracting good correspondences using the nearest neighbor in feature space. We provide a brief overview in the next section.

Functional Maps The functional map (fmap) framework was used for the first stage of our **NCP-UN** algorithm. For this, we follow the general strategy of recent fmap-based techniques [11, 12, 39, 15, 19], as follows: given source and target shapes $\mathcal{M}$ and $\mathcal{N}$, represented as either triangular meshes or point clouds, having m and n vertices respectively, we pre-compute their Laplace-Beltrami operator [89], and store their first k eigenfunctions in the matrices $\Phi_{\mathcal{M}} \in \mathbb{R}^{m\times k}$ and $\Phi_{\mathcal{N}} \in \mathbb{R}^{n\times k}$ respectively. Using a Siamese network $\mathcal{F}_\theta$, we compute for each shape a d-dimensional descriptor $\mathcal{F}_\theta(\mathcal{M}) = \mathbf{F} \in \mathbb{R}^{m\times d}$ and $\mathcal{F}_\theta(\mathcal{N}) = \mathbf{G} \in \mathbb{R}^{n\times d}$ respectively. These descriptors are then projected to the spectral domain to form the spectral features $\mathbf{A} = \Phi_{\mathcal{M}}^{\dagger}\mathbf{F}$ and $\mathbf{B} = \Phi_{\mathcal{N}}^{\dagger}\mathbf{G}$ ($\bullet^{\dagger}$ is the Moore pseudo-inverse). A functional map is then computed by solving the following linear system:

$$C_{opt} = \arg\min_{C} \|C\mathbf{A} - \mathbf{B}\| + \lambda\|C\Delta_{\mathcal{M}} - \Delta_{\mathcal{N}}C\|. \tag{1}$$

where $\Delta_{\mathcal{M}}, \Delta_{\mathcal{N}}$ are diagonal matrices of Laplace-Beltrami eigenvalues of the corresponding shapes and λ is a scalar hyper-parameter.

Following the unsupervised literature [15, 39, 19], the Siamese network $\mathcal{F}_\theta$ is trained by imposing structural properties on the fmap C such as bijectivity and orthogonality on the shape pairs in the training set. In fact, given the fmap $C_{\mathcal{MN}}$ from $\mathcal{M}$ to $\mathcal{N}$, and $C_{\mathcal{NM}}$ from $\mathcal{N}$ to $\mathcal{M}$, the bijectivity loss is formulated as $\|C_{\mathcal{MN}}C_{\mathcal{NM}} - \mathbb{I}_k\|_2^2$, and the orthogonality loss is $\|C_{\mathcal{MN}}^{\top}C_{\mathcal{MN}} - \mathbb{I}_k\|_2^2$. $\mathbb{I}_k$ denotes the identity matrix of size k and $\bullet^{\top}$ is the matrix transpose operator.

Feature Learning The goal of feature learning is to learn robust descriptors that can allow direct nearest-neighbor matching in the descriptor space. In this work, we use two losses: PointInfoNCE and the LIE loss.

PointInfoNCE [72] is a contrastive loss such that, given a set of matched points $\mathcal{P}$, and two features of dimension s, it is formulated as follows:

$$\mathcal{L}_{\text{NCE}} = -\sum_{(i,j)\in\mathcal{P}} \log \frac{\exp(d(\mathbf{F}_i, \mathbf{G}_j)/\tau)}{\sum_{(\cdot,k)\in\mathcal{P}} \exp(d(\mathbf{F}_i, \mathbf{G}_k)/\tau)} \tag{2}$$

$$d(\mathbf{F}_i, \mathbf{G}_j) = \|\mathbf{F}_i - \mathbf{G}_j\|_2^2 \tag{3}$$

where τ is a temperature parameter, and $d(\cdot,\cdot)$ is the Euclidean distance between the two features. In all our experiments, we took $\tau = 0.07$. The purpose of this loss is to force the distance between the features of the matched points to be minimized, while this distance must be maximized between the unmatched points. The NCE loss is applied to each point individually and thus cannot penalize the overall consistency of the matches.

To remedy this, especially when the number of vertices of the shapes is moderate, as is the case for sparse point clouds, we use the LIE loss introduced in [75]. Given the extracted features $\mathbf{F}$ and $\mathbf{G}$, and the coordinate *xyz* of the shape $\mathcal{N}$ represented by the matrix $\mathbf{N} \in \mathbb{R}^{n\times 3}$, we first compute the soft correspondences matrix $S_{\mathcal{MN}}$, and then formulate the LIE loss as follows:

$$\mathcal{L}_{\text{LIE}} = \|S_{\mathcal{MN}}\mathbf{N} - \Pi^{gt}_{\mathcal{MN}}\mathbf{N}\|_2^2, \tag{4}$$

$$(S_{\mathcal{MN}})_{ij} = \frac{\exp(-\|\mathbf{F}_i - \mathbf{G}_j\|_2)}{\sum_{k=1}^{n} \exp(-\|\mathbf{F}_i - \mathbf{G}_k\|_2} \tag{5}$$

where $\Pi^{gt}_{\mathcal{MN}}$ is the ground truth p2p correspondence matrix. This loss forces the soft correspondences matrix to be as close as possible to the ground truth map, by forcing their action (pull-back) on the shape coordinates, thus taking into account the geometry of the shape. Indeed, erroneous predictions that are geometrically close to the ground truth are penalized less than those that are far from it in terms of L2 distance.

B Implementation details

In Section 3.1 in the main text, we used a randomly initialized DiffusionNet [19] network to predict features on the test set of FAUST-Remeshed and SCAPE-Remeshed datasets [33]. For that, we used the publicly available implementation of DiffusionNet released by the authors [1]. Unless specified otherwise, all our experiments on 3d-triangular meshes use 4 DiffusionNet blocks of width 128, the input to the network is the XYZ coordinates of the shape. For the competing features, both the Heat Kernel Signature (HKS) [69] and the Wave Kernel Signature (WKS) [70] were sampled at 100 values of energy t, logarithmically spaced in the range proposed in their respective original papers. SHOT descriptors [64] are 352-dimensional, and we used the implementation provided by the PCL library [90]. For the Laplace-Beltrami computation, we used the discretization introduced in [89] for both 3D meshes and point clouds.

In Section 3.2 of the main text, we train a DiffusionNet to produce feature embeddings that will induce the maps used for supervision, using the NCE loss. For this experiment and all the following learning experiments, ADAM optimizer [91] was used with a learning rate of 0.001.

In Section 5.1.1 and 5.2 of the main text, we applied our **NCP-UN** algorithm on the KeyPointNet [57] and the PartNet [76] datasets. For the first stage, we used the unsupervised geometric functional map from [19] but using PointMLP [74] as a feature extractor, instead of DiffusionNet, as it is better suited to the point cloud context. We used the default segmentation configuration provided by the authors [2]. In Eq. (1), we take $\lambda = 0$. The network is trained using only the bijectivity loss presented

[1] `https://github.com/nmwsharp/diffusion-net`

[2] `https://github.com/ma-xu/pointMLP-pytorch`

in Appendix A. For the second stage, we used the LIE loss, and we trained a randomly initialized PointMLP, which has the same architecture as the first stage, to produce feature embeddings that induce the input maps.

In Section 5.1.2 of the main text, **NCP-UN** was applied on the SMAL [77, 36] and SHREC'20 [78] datasets. The first stage was performed using the unsupervised geometric functional map [19] with the DiffusionNet backbone, while applying the bijectivity and orthogonality losses, as described above in Appendix A. The second stage was performed using the LIE loss and the same backbone as the first stage. P2P maps were extracted using either the nearest neighbor in the space of features or using the functional map pipeline. In fact, given two feature embedding for two shapes $\mathcal{M}$ and $\mathcal{N}$, we compute the functional map $C_{\mathcal{MN}}$ from $\mathcal{M}$ to $\mathcal{N}$ using Eq. (1), and then convert it to a p2p map $T_{fmap} : \mathcal{N} \to \mathcal{M}$ using (borrowing the same notation from Appendix A):

$$T_{fmap}(y) = \arg\min_x \|(\Phi_{\mathcal{N}})_y - (\Phi_{\mathcal{M}} C_{\mathcal{MN}}^{\top})_x\| \tag{6}$$

For the test time optimization experiment in Section 5.1.2 of the main text, after extracting the maps predicted by the first stage, we construct pairs between all the shapes of the test set, and for each pair, we train a randomly-initialized DiffusionNet network using the NCE loss and Adam optimizer, to produce feature embedding that induces the map from stage 1. We stop the training when the cyclic loss (see Section 4.2 of the main text) stops improving giving a patience period of 100 optimization iterations.

In all the experiments of Section 5 of the main text, except for test time optimization, data augmentation was used. In particular, we augment the training data on the fly by randomly rotating the input shapes, applying random scaling in the range [0.9, 1.1], and jittering the position of each point by Gaussian noise with zero mean and 0.01 standard deviation.

Computational specifications All our experiments are executed using Pytorch [92], on a 64-bit machine, equipped with an Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz and a RTX 2080 Ti Graphics Card. For all competing methods, we use the original code released by the authors and apply the best parameters reported in the respective papers. As mentioned in the main paper, we will release our complete implementation to ensure the full reproducibility of all of our results.

C Implementation Details on FSKD

Below we provide the implementation details on our Few-Shot Keypoint Detection method (FSKD), described in Section 4.3 of the main manuscript. As mentioned in that section, **FSKD** is composed of three steps: **1.** Detection of potential keypoints by transferring keypoints from labeled shapes, **2.** Filtering to remove keypoints that are likely not to exist on the target shape, **3.** Combination: merge transferred keypoints if multiple points on the target shape are assigned to the same keypoint ID.

Step 1. is done using our established maps predicted by **NCP-UN**.

For **step 2.**, we compute the cycle consistency loss of transferred keypoints and only keep the ones that are below a predetermined threshold. I.e., given a pair of shapes $\mathcal{M}, \mathcal{N}$, and two maps between them $\Pi_{\mathcal{MN}}$ and $\Pi_{\mathcal{NM}}$ computed via nearest neighbor matching between their feature embeddings, and represented as binary matrices, the cycle consistency loss of keypoints i is computed as $l_i = \|(X_{\mathcal{N}})_i - (\Pi_{\mathcal{NM}}\Pi_{\mathcal{MN}}X_{\mathcal{N}})_i\|_F^2$, where $X_{\mathcal{N}}$ is the matrix of XYZ coordinates of the source shape $\mathcal{N}$. If l_i is bigger than a predefined threshold ν, the keypoint i is considered not to exist on the target shape. In our experiments, we take $\nu = 0.05$.

Concerning **step 3.**, we perform a spatially weighted average of the different filtered keypoints if there are many. We associate for each keypoint i the weight $w_i = \frac{D_i}{\sum_j D_j}$, where $D_i = \exp(-\frac{l_i}{\sigma})$. We use $\sigma = 0.01$.

D Proof of smoothness of maps produced by smooth networks

In Section 3.2 of the main text, we briefly mentioned a theorem that states that maps based on the nearest neighbor between smooth features are smooth. In this section, we formally restate it and provide proof.

Theorem 1. *Let $\mathcal{M}$ and $\mathcal{N}$ be two compact smooth surfaces (smooth manifolds of dimension 2). Let $\mathbf{M}$ and $\mathbf{N}$ be their embeddings in $\mathbb{R}^d$, given by some functions: $\psi : \mathcal{M} \to \mathbb{R}^d$ and $\phi : \mathcal{N} \to \mathbb{R}^d$, so that $\mathbf{M} = \psi(\mathcal{M})$ and $\mathbf{N} = \phi(\mathcal{N})$. Suppose that ψ and ϕ that are both smooth and injective. Then up to arbitrarily small perturbations of ϕ, ψ, the map $T_{nn} : \mathcal{M} \to \mathcal{N}$ given by $T_{nn}(x) = \arg\min_{y \in \mathcal{N}} \|\psi(x) - \phi(y)\|$ must be smooth up to sets of measure 0 on $\mathcal{M}$.*

Proof. First, note that the distance function to an embedded manifold is smooth almost everywhere. Indeed, it is well-known that if $\mathbf{N}$ is a C^k-continuous manifold embedded in $\mathbb{R}^d$, then the distance function to $\mathbf{N}$ must be at least C^k continuous on the complement of the medial axis of $\mathbf{N}$. I.e., let $d_{\mathbf{N}} : \mathbb{R}^d \to \mathbb{R}^d$ be given by $d_{\mathbf{N}}(x) = \arg\min_{y \in \mathbf{N}} \|x - y\|$. Let $\mathrm{cut}(\mathbf{N})$ denote the medial axis of $\mathbf{N}$, which is defined as the set of points in $\mathbb{R}^d$ with more than one nearest neighbor to $\mathbf{N}$. I.e., $\mathrm{cut}(\mathbf{N}) = \{\mathrm{x} \in \mathbb{R}^{\mathrm{d}} \mid \exists\, \mathrm{y}_1, \mathrm{y}_2 \in \mathbf{N}, \mathrm{y}_1 \neq \mathrm{y}_2, \mathrm{s.t.} \|\mathrm{x} - \mathrm{y}_1\| = \|\mathrm{x} - \mathrm{y}_2\| = \min_{\mathrm{y} \in \mathbf{N}} \|\mathrm{x} - \mathrm{y}\|\}$. Then, $d_{\mathbf{N}}$ is at least C^k continuous on $\mathbb{R}^d \backslash \mathrm{cut}(\mathbf{N})$ (see Lemma 2.5 in [93], and [94, 95] for this and related results).

It remains to prove that up to arbitrarily small perturbations of ϕ and ψ the intersection between $\mathbf{M}$ and $\mathrm{cut}(\mathbf{N})$ has measure zero on $\mathbf{M}$. For this, we first use the fact that the medial axis of compact subanalytic manifolds is also subanalytic [96]. This means that the medial axis can be stratified (decomposed into a finite union of submanifolds of dimension $d - 1$ in $\mathbb{R}^d$). Since (by Stone-Weierstrass's theorem) subanalytic manifolds are dense in the space of smooth manifolds, up to an arbitrarily small perturbation of ϕ, $\mathbf{N}$ is subanalytic. Finally, the intersection between $\mathbf{M}$ (which is an embedded manifold of dimension 2) and any manifold of dimension $d - 1$ by Thom's transversality theorem [97, 98], must generically be measure zero on $\mathbf{M}$ and thus on $\mathcal{M}$. □

E Verification of assumptions of Theorem 1

The main motivation behind Theorem 1 is to highlight the fact that, given smooth feature embeddings, if we add the injectivity condition, the maps extracted using the nearest neighbors in the feature space tend to be smooth. As smoothness is a generally desirable property, and the frequency bias shown in prior works, such as [23] suggests that neural networks are biased towards low frequency (and thus smooth) functions, we provide this result as a partial explanation for the Neural Correspondence Prior, which we have observed, for the first time, in our work.

Regarding injectivity, although such a property is not trivial, and might need to be specifically enforced (using, for example, invertible networks [99]). However, we observe that when training a network using contrastive learning, such as NCE loss, the latter forces the networks to produce embeddings that are unique for each point, in order to minimize the NCE loss, which can be considered a form of injectivity.

We include in this section the results of an experiment checking the smoothness and injectivity of a randomly initialized network. For this, we performed the following experiment. We first computed the embeddings produced by a randomly initialized DiffusionNet network for the 20 test shapes in the FAUST-Remeshed [33] dataset.

To evaluate the smoothness of the embedding, we computed the standard Dirichlet energy of both the embedding produced by the network and the original embedding of the shape in (the 3D coordinates of the shape's vertices), using the following formula:

$$E_{Dirichlet}(G) = \frac{1}{n} \sum_{i=1}^{n} \frac{G_i^\top W G_i}{G_i^\top A G_i} \tag{7}$$

where G is the considered embedding of dimension n, while W and A are, respectively, the standard stiffness and mass matrices, computed using the classical cotangent discretization scheme of the Laplace-Beltrami operator [100]. We computed the Dirichlet energy over the whole test set and found it equal to **47.2** (the average) for coordinate functions (original shape embeddings in), while it is equal to **13.1** for embeddings produced by the network. This shows that the latter is smoother than embeddings in the original space.

For injectivity, we computed for each point of the embedding, the distance to its nearest neighbor, and took the minimum across all points of a shape. To make the comparison between the original

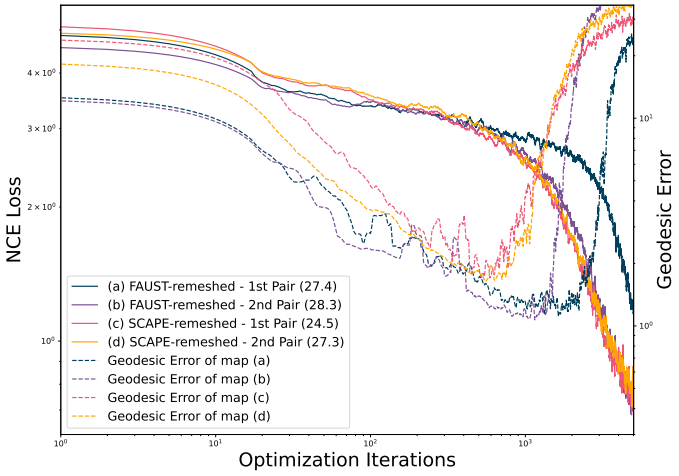

Figure 6: Learning curves showing the NCE loss for different shape pairs from FAUST-Remeshed and SCAPE-Remeshed datasets. Numbers in parentheses represent the geodesic error of the input maps. Observe how the effect of NCP doesn't depend on the shape pair.

embedding and the embedding produced by the network fair, we normalized both embeddings to the unit sphere. We find that on average, the minimum distance between points in the original 3D space is **0.0004**, while for the feature embedding, given by the network it is equal to **0.0015**. This shows that the network is injective and that distances between points are larger than in the original domain.

F Neural Correspondence Prior

In section 3 of the main text, we demonstrated the effect of Neural Correspondence Prior (NCP), on a pair of shapes, using the DiffusionNet network. The objective of this section is to show that the NCP is independent of the choice of the shape pair, the choice of the p2p loss, the choice of the architecture, and finally the choice of the first stage.

F.1 Independence from shape pairs

In order to show that the NCP demonstrated in Figure 1 of the main text does not depend on the chosen shape, we redid the same experiment using new random pairs from FAUST-Remeshed and SCAPE-Remeshed datasets [33]. Following the same setup of Section 3.2 of the main text, given a pair of shapes, we corrupt the ground truth map between them with 50% noise, and then train a randomly initialized DiffusionNet to produce feature embeddings that overfit the noisy map. Examples of four different pairs are shown in Fig. 6. We see that the same effect is always present, i.e., the network resists overfitting the noisy maps (high noise impedance), and the intermediate maps during optimization are of high quality, compared to the noisy maps used as supervision.

F.2 Independence from the Loss Function and Network Architecture

In order to check the dependence of NCP on both the network architecture and on the p2p loss used in stage 2 of **NCP-UN**, the following experiment was performed on the Chair subset of the KeyPointNet [57] dataset. Since the latter doesn't provide dense ground truth p2p maps between the shapes, we took the p2p maps produced by the first stage of our algorithm (see Section 5.1.1 of the main text), and perturbed them with noise. These noisy maps are used as inputs to the second stage of the **NCP-UN** algorithm. Differently from the experiment performed in Section 5.1.1 of the main text, here, we choose different network architectures and losses for training the second stage. In particular, we consider the NCE loss, the LIE loss, and the supervised FMAP loss from [11]. Concerning the network architectures, we considered the ResidualMLP network introduced in [13], PointNet++ [21],

Table 4: Results of **NCP-UN** on the Chair subset of the KeyPointNet dataset [57] with different losses and network architectures. It appears that multiple network architectures and losses improve the maps produced by the first stage, proving that NCP is not tied to a single method choice

Metric	Input Maps	PointMLP + LIE	PointMLP + NCE	PointMLP + FMAP	DGCNN + LIE	PointNet++ + LIE	ResidualMLP + LIE
Geodesic Error (×100)	9.5	5.1	5.1	5.2	5.0	5.3	8.1
Map Smoothness	234.9	10.3	10.1	9.9	11.3	9.7	8.3

Table 5: Results of **NCP-UN** on the SMAL dataset using multiple methods for Stage 1. Values are mean geodesic error ×100 on unit-area shapes. It can be seen that the NCP effect still applies despite the chosen method for Stage 1

Stage / Method	NeuroMorph [83]	Smooth Shells [18]	Deep Shells [17]	Unsup GeomFMaps [19]
Stage 1	23.1	16.3	15.2	7.8
Stage 2	7.2 (+68%)	8.1 (+50%)	7.3 (+52%)	5.8 (+25%)

and DGCNN [20]. For the network architectures, we used the official implementation provided by the authors.

In addition to measuring the geodesic error produced by the maps predicted by the second stage, we also measure their smoothness. Given two shapes $\mathcal{M}, \mathcal{N}$, and a map between them $T_{\mathcal{MN}}$ represented as a binary matrix $\Pi_{\mathcal{MN}}$, we compute the smoothness of the latter based on the Dirichlet energy using the following (see [100] for more details):

$$E_{smoothness}(T_{\mathcal{MN}}) = \sum_{(u,v)\in\mathcal{E}_{\mathcal{M}}} w_{uv} \|\psi_{\mathcal{MN}}(u) - \psi_{\mathcal{MN}}(v)\|_2^2 \tag{8}$$

$$\psi_{\mathcal{MN}} = \Pi_{\mathcal{MN}} X_{\mathcal{N}} \tag{9}$$

where $X_{\mathcal{N}}$ is the matrix of XYZ coordinates of shape $\mathcal{N}$, $\mathcal{E}_{\mathcal{M}}$ is the set of all the edges of the triangular mesh, and w_{uv} are the stiffness weights of the cotangent Laplacian [100] for shape $\mathcal{M}$.

Results of this experiment are summarised in Tab. 4. It can be seen that all the losses and network architectures perform well and manage to improve the noisy input maps. One can also notice the smoothing effect of NCP. In fact, the second stage of the network produces feature embeddings that result in p2p maps that are not only geometrically correct but also remove noise and outliers from the input maps. It is worth noting that although the ResidualMLP architecture is weak, since it is applied on each vertex individually, and has no global shape awareness, we note that even if the geodesic error is bad, the network tends to produce smooth embeddings.

F.3 Independence from the first stage

As we stated in the main paper, our **NCP-UN** algorithm is independent from the first stage, i.e the method used to extract the artifact-laden p2p is not crucial, and the NCP effect still applies. To demonstrate this, we followed the same setup of section 5.1.2 of the main text, and we perform the same experiment on the SMAL dataset but using different methods for stage 1. In particular, we consider as stage 1 the Unsupervised geometric functional map (Unsup GeomFMaps) [19], Smooth Shells [18], Deep Shells [17] and NeuroMorph [83]. Results are summarized in Tab. 5. As can be seen, despite the method used in the first step, the NCP still applies, and a significant improvement in results is observed, up to 68% improvement.

G Shape Matching On Man-Made Data

In Section 5.1.1 of the main text, we applied our **NCP-UN** algorithm on the KeyPointNet dataset, and only provided quantitative results for 8 classes due to the page limit. We provide in Fig. 7 quantitative results for all the 16 classes, in addition to the results of the first stage of our algorithm, i.e the unsupervised method described above in Appendix B. For the bottle, cap, knife, and skate categories, we only compared them to the best-performing baseline. It can be seen that our method achieve state-of-the-art results in 13 out of 16 classes on this benchmark, with an impressive improvement in

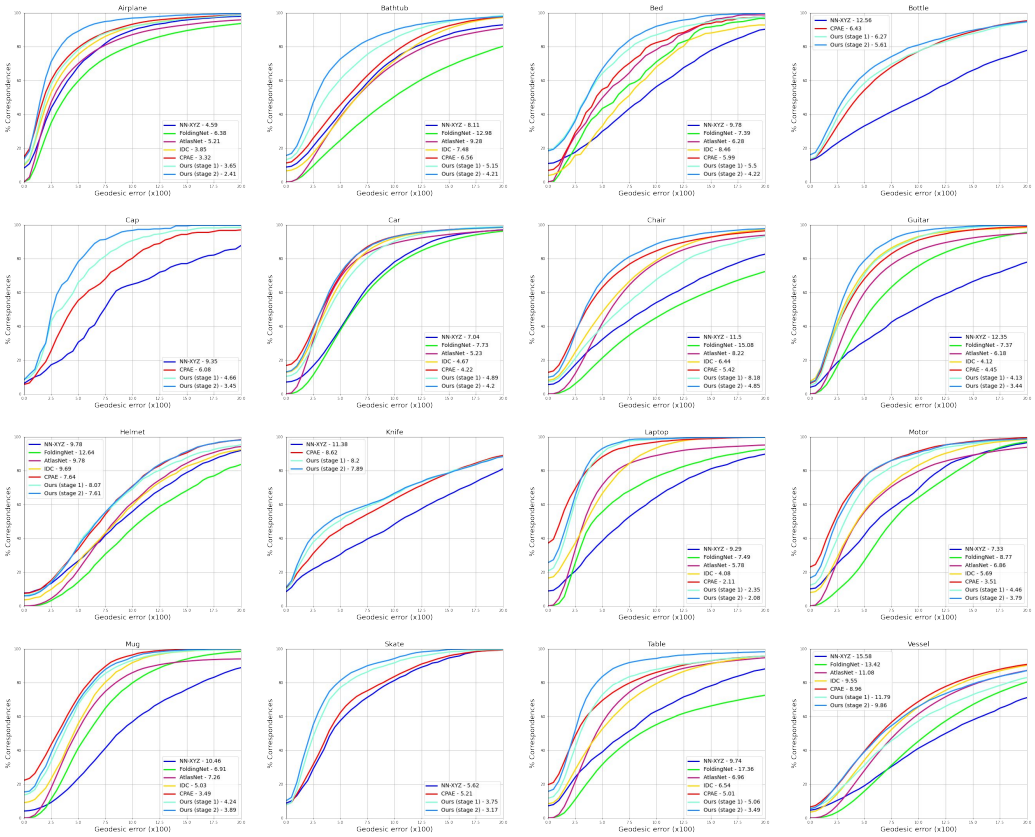

Figure 7: Correspondence accuracy on the KeyPointNet dataset. It can be seen that the second stage of our algorithm always improves upon the first stage due to the NCP effect.

some classes such as skate and table. We also see that the second stage of **NCP-UN** always improves the result provided by the first stage, which again demonstrates the role of the NCP effect, without which the SOTA result would not be achieved, see for example the chair category.

Additionally, we provide in Fig. 8 some qualitative results, showing the p2p maps produced by both stages of **NCP-UN**, visualized using texture transfer, as well as the usage of these maps to transfer keypoints between shapes. It can be seen that the keypoint transferred by the maps of stage 2 are more accurate.

H Shape Matching On Non-Rigid Near-Isometric Data

In our work, we focus on difficult non-rigid non-isometric datasets, where existing methods tend to fail. This is because on near-isometric datasets such as FAUST Remeshed or SCAPE Remeshed [33], current unsupervised methods can exploit the assumption of near-isometry and achieve good results.

Nevertheless, we include a comparison of our method on the FAUST-Remeshed and SCAPE-Remeshed datasets, using the same train/test split used in all previous works (e.g. [11]). The notation X on Y means the method is trained on X and tested on Y.

As input for our Stage 2, we use WSupFMNet + DiffusionNet [19] (referred to as stage 1 in the table), which is the same method used in the main manuscript for all our experiments. For our method (Stage 2), we use the same implementation as the one used for the SMAL dataset, described in Section 5.1.2 of the main manuscript.

As shown in Tab. 6, even on a near-isometric dataset, our Stage 2 improves upon the initial maps in all categories, by **18.4%** on average. Nevertheless, we remark that in such settings it is more

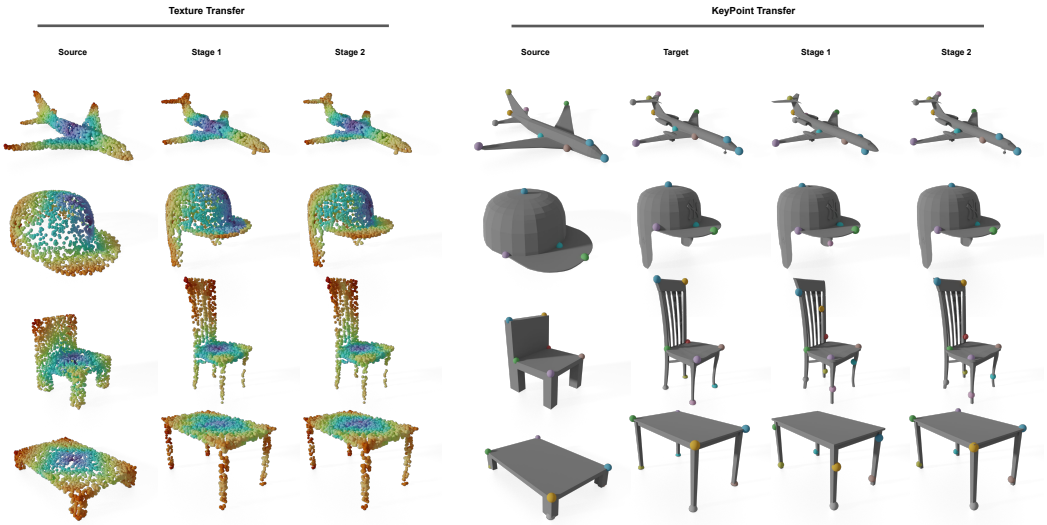

Figure 8: Qualitative results on the KeyPointNet dataset using four categories: airplane, cap, chair, and table. Both the point-to-point map as well as the keypoint transfer are presented. Each row contains the p2p maps, ground truth keypoint annotations, and predictions made by the first and the second stage of our algorithm.

Table 6: Results of **NCP-UN** on the FAUST Remeshed (**FR**) and SCAPE Remeshed (**SR**) datasets. Values are mean geodesic error ×100 on unit-area shapes. It can be seen that the NCP effect still applies even in the case of near isometric shapes

Method	FR on FR	SR on SR	FR on SR	SR on FR
Stage 1	3.8	4.4	4.8	3.6
Stage 1 + ZoomOut	1.9	2.6	2.7	1.9
NCP (Ours - Stage 2)	**3.0**	**3.5**	**4.2**	**2.9**
NCP (Ours - Stage 2) + ZoomOut	1.9	2.4	2.6	1.9

advantageous to use specialized methods, such as ZoomOut [101], that directly exploit the near-isometry assumption.

I Societal Impact

Efficient methods for shape matching and analysis have immediate impact in many areas of science and engineering from medical imaging (for instance for detecting anomalies, and performing follow-up analysis) to shape recognition and classification in areas such as computational biology, archaeology and paleontology to name a few. Our approach can immediately be adapted in such diverse scenarios, due to its strong generalization power, and to its generic unsupervised nature, especially in domains where acquiring data is easy, but labeling it is very expensive, e.g in structural or molecular biology. Our work also opens up important avenues for future research, as it can enable geometric deep learning methods without the need to label large-scale datasets, thus potentially allowing small labs to conduct research in this area without the need to hire many annotators, which is an expensive task. Finally, our method paves the way for more accurate results, helping to improve our understanding in many fields, such as biology where shape matching techniques are used to analyze gene expression patterns to understand the cause of many human syndromes [102]. Since our method attempts to solve a fundamental problem in computer graphics and computer vision, we do not expect negative results. However, one should note that highly accurate shape correspondence methods might have possibly problematic uses, e.g., in surveillance applications, although we advocate against such uses of our technique.