G-MSM: Unsupervised Multi-Shape Matching with Graph-based Affinity Priors

Marvin Eisenberger¹, Aysim Toker¹, Laura Leal-Taixé¹,†, Daniel Cremers¹,²,³
¹TU Munich, ²Munich Center for Machine Learning, ³University of Oxford

Abstract

We present G-MSM (Graph-based Multi-Shape Matching), a novel unsupervised learning approach for non-rigid shape correspondence. Rather than treating a collection of input poses as an unordered set of samples, we explicitly model the underlying shape data manifold. To this end, we propose an adaptive multi-shape matching architecture that constructs an affinity graph on a given set of training shapes in a self-supervised manner. The key idea is to combine putative, pairwise correspondences by propagating maps along shortest paths in the underlying shape graph. During training, we enforce cycle-consistency between such optimal paths and the pairwise matches which enables our model to learn topology-aware shape priors. We explore different classes of shape graphs and recover specific settings, like template-based matching (star graph) or learnable ranking/sorting (TSP graph), as special cases in our framework. Finally, we demonstrate state-of-the-art performance on several recent shape correspondence benchmarks, including real-world 3D scan meshes with topological noise and challenging inter-class pairs.¹

1. Introduction

Shape matching of non-rigid object categories is a central problem in 3D computer vision and graphics that has been studied extensively over the last few years. Especially in recent times, there is a growing demand for such algorithms as 3D reconstruction techniques and affordable scanning devices become increasingly powerful and broadly available. Classical shape correspondence approaches devise axiomatic algorithms that make specific assumptions about the resulting maps, such as near-isometry, area preservation, approximate rigidity, bounded distortion, or commutativity with the intrinsic Laplacian. In contrast, real-world scan meshes are often subject to various types of noise, including topological changes [16, 33], partial views [2], general non-isometric deformations [17, 65], objects in clutter [12], and varying data representations [57]. In this work, we address several of the aforementioned challenges and demonstrate that our proposed method achieves improved stability for a number of 3D scan mesh datasets.

The majority of existing deep learning methods for shape matching [2, 15, 20, 21, 24, 39, 51, 56] treat a given set of meshes as an unstructured collection of poses. During training, random pairs of shapes are sampled for which a neural network is queried and a pairwise matching loss is minimized. While this approach is straightforward, it often fails to recognize commonalities and context-dependent patterns

¹ Currently at NVIDIA

¹ Our implementation is available under the following link: https://github.com/marvin-eisenberger/gmsm-matching

Figure 1. For a given collection of 3D meshes \(\{\chi^i\}_{1 \leq i \leq N}\), (i) our method constructs, in a fully unsupervised manner, a shape graph \(G\) which approximates the underlying shape data manifold. (ii) Its edge weights (affinity scores) are derived from a putative pairwise correspondence loss signal. (iii) During training, we enforce cycle-consistency by propagating maps along shortest paths in the graph \(G\). As shown for the sample pair above \((\chi^{(1)}, \chi^{(2)})\), the resulting multi-matching \(\Pi^{(1,3)} \circ \Pi^{(3,2)}\) is significantly more accurate than the pairwise map \(\Pi^{(1,2)}\).
which only emerge from analyzing the shape collection as a whole. Not all samples of a shape collection are created equal. In most cases, some pairs of poses are much closer than others. Maps between similar geometries are inherently correlated and convey relevant clues to one another. This is particularly relevant for challenging real-world scenarios, where such redundancies can help disambiguate noisy geometries, non-isometric deformations, and topological changes. The most common approach of existing multi-matching methods is to learn a canonical embedding per pose, either in the spatial [10] or Laplace-Beltrami frequency domain [26,28]. This incentivizes the resulting matches to be consistent under concatenation. However, such approaches are in practice still trained in a fully pairwise manner for ease of training. Furthermore, relying on canonical embeddings can lead to limited generalization for unseen test poses. Concrete approaches often assume a specific mesh resolution and nearly-isometric poses [10], or require an additional fine-tuning optimization at test time [26, Sec. 5].

Rather than interpreting a given training set as a random, unstructured collection of shapes, our approach explicitly models the underlying shape manifold. To this end, we define an affinity graph \( G \) on the set of input shapes whose edge weights (i.e. affinity scores) are informed by the outputs of a pairwise matching module. We then devise a novel adaptive multi-matching architecture that propagates matches along shortest paths in the underlying shape graph \( G \). The resulting maps are topology-aware, i.e., informed by geometries from the whole shape collection. An example is shown in Figure 1, where the multi-matching \( \Pi^{(1,3)} \circ \Pi^{(3,2)} \) obtained by our approach is significantly more accurate than the naive, pairwise map \( \Pi^{(1,2)} \). During training, we promote cycle-consistency of shortest paths in the shape graph. In summary, our contributions are as follows:

1. Introduce the notion of an edge-weighted, undirected shape graph \( G \) to approximate the underlying data manifold for an unordered collection of 3D meshes.

2. Propose a novel, adaptive multi-shape matching approach that enforces cycle-consistency for optimal paths in the shape graph \( G \) in a self-supervised manner.

3. Demonstrate state-of-the-art performance for a range of challenging non-rigid matching tasks, including non-isometric matching due to topological noise [16,33] and inter-class pairs [17,65].

2. Related work

Axiomatic correspondence methods Shape matching is an extensively studied topic with a variety of different approaches and methodologies. We summarize references relevant to our approach here and refer to recent surveys [54,60] for a more complete picture. Classical methods for non-rigid matching often devise optimization-based approaches that minimize some type of distortion metric [7,18,49,62,64]. A common prerequisite is the extraction of hand-crafted local descriptors that are approximately preserved under non-rigid shape deformations. Common definitions include histogram-based statistics [59] or fully intrinsic features based on the eigenfunctions of the Laplace-Beltrami operator [3,53,58]. Over the last few years, functional maps [44] have become a central paradigm in shape matching. The core idea is to reframe the pairwise matching task from functions (points to points) to functionals (functions to functions). There are several extensions of the original framework to allow for partial matching [35,36,50], orientation preservation [48], iterative map upsampling [19,42] and conformal maps [14]. Our approach utilizes functional maps as a fundamental building block within the differentiable matching layer.

Learning-based methods More recently, several approaches emerged that aim at extending the power of deep feature learning to deformable 3D shapes. Many such methods fall under the umbrella term ‘geometric deep learning’ [9], with analogous applications on different classes of non-Euclidean data like graphs or general manifold data. One class of approaches are charting-based methods [6,40,43,46,52] which imitate convolutions in Euclidean space with parameterized, intrinsic patch operators. Likewise, [57] proposed a learnable feature refinement module based on intrinsic heat diffusion.

The pioneering work of [34] proposes a differentiable matching layer based on functional maps [44], in combination with a deep feature extractor with several consecutive ResNet layers [25]. Numerous extensions of this paradigm were proposed over the last few years to allow for unsupervised loss functions [24,51], learnable basis functions [39], point cloud feature extractors [15,26,56] or partial data [2]. Similarly, DeepShells [21] learns functional maps in an end-to-end trainable, hierarchical multi-scale pipeline. We adapt parts of its differentiable matching layer in our network. Other approaches [4,23,38] learn correspondences for a specific class of deformable objects by including additional domain knowledge, like a deformable human model [37]. Finally, [20] jointly learns to predict correspondences and a smooth interpolation between pairs of shapes.

Multi-shape matching Classical axiomatic multi-shape matching approaches devise optimization-based pipelines that enforce cycle-consistent maps. Specific solutions include semidefinite programming [27], convex relaxations of the corresponding quadratic assignment problem [30], graph cuts [55], as well as evolutionary game theory [11]. Such optimization-based approaches are computationally costly and therefore limited to matching sparse landmarks. Furthermore, there are a number of optimization frame-
works that compute synchronized, cycle-consistent functional maps [22, 28, 29]. Notably, such approaches are often limited to nearly-isometric poses [22, 29] or require high-quality initializations [28]. More recent learning-based approaches promote cycle-consistency by predicting a canonical embedding for each observed pose [10, 26]. However, obtaining stable embeddings is often difficult when generalizing to unseen test poses. Moreover, such approaches assume a specific mesh resolution and nearly-isometric poses [10] or require an additional fine-tuning optimization at test time to obtain canonical embeddings [26, Sec. 5].

3. Method

3.1. Problem formulation

In the following, we consider a collection of 3D shapes \( \mathcal{S} = \{ \mathcal{X}^{(i)}, \ldots, \mathcal{X}^{(N)} \} \) from non-rigidly deformable shape categories. Each such shape \( \mathcal{X}^{(i)} \) is a discretized approximation of a 2D Riemannian manifold, embedded in \( \mathbb{R}^3 \). Specifically, we define \( \mathcal{X}^{(i)} = (V^{(i)}, T^{(i)}) \), where \( V^{(i)} \in \mathbb{R}^{m \times 3} \) and \( T^{(i)} \subset V^{(i)} \times V^{(i)} \times V^{(i)} \) are sets of vertices and triangular faces, respectively. The goal is then to construct an algorithm that computes dense correspondence mappings \( \Pi^{(i,j)} \) between any two shapes \( \mathcal{X}^{(i)} \) and \( \mathcal{X}^{(j)} \) from the shape collection \( \mathcal{S} \). Specifically, such correspondences are represented by sparse assignment matrices \( \Pi^{(i,j)} \in \{0,1\}^{m \times n} \), where \( \Pi^{(i,j)}_{1:n,1:m} = 1 \) indicates a match between the \( i \)-th vertex of \( \mathcal{X}^{(i)} \) and the \( j \)-th vertex of \( \mathcal{X}^{(j)} \).

Scope Our method is unsupervised and thereby requires no additional inputs, like landmark annotations or ground-truth correspondences, beyond the raw input geometries \( \mathcal{X}^{(i)} \). Following similar approaches in this line of work [4, 20, 39, 56], we assume that the shapes \( \mathcal{X}^{(1)}, \ldots, \mathcal{X}^{(N)} \) have an approximately canonical orientation. In the literature, this setting is commonly referred to as ‘weakly supervised’, see [56] and later [20]. Existing approaches often make additional assumptions about the input data \( \mathcal{S} \), focusing on nearly-isometric correspondences [34, 51], maps with bounded distortion [20, 21] or partial views of the same non-rigid object [2, 50]. Others specialize in distinct classes of shapes like deformable human bodies [4, 23, 38]. In contrast, we demonstrate in our experiments that our proposed multi-matching approach excels at a broad range of challenging settings, including non-isometric pairs, poses with topological noise from self-intersections, and inter-class matching.

3.2. Network architecture

We now define the neural network architecture that forms the basis of our proposed approach. It consists of three separate components \( \textbf{I-III} \), see also Figure 2 for an overview. The first two modules are standard components found in most learning-based shape matching approaches [2, 15, 20, 21, 24, 39, 51, 56], namely a learnable feature backbone \( \Phi \) and a differentiable, pairwise matching layer \( \Phi \). We briefly outline these here and provide additional details in Appendix A.2. The multi-matching architecture \( III \) is introduced in Section 3.3.

Feature extractor The first component \( I \) of our model is a standard, learnable feature extraction backbone for representation learning, defined as

\[ \Phi_{\text{feat}} : \mathcal{X}^{(i)} \mapsto \mathbf{F}^{(i)} \in \mathbb{R}^{m \times l}. \]  

(1)

For a given input shape \( \mathcal{X}^{(i)} = (V^{(i)}, T^{(i)}) \), the mapping \( \Phi_{\text{feat}} \) produces an \( l \)-dimensional feature embedding \( \mathbf{F}^{(i)} \) per vertex \( V^{(i)} \in \mathbb{R}^{m \times 3} \). While other choices are possible, we base \( \Phi_{\text{feat}} \) on the off-the-shelf feature backbone DiffusionNet [57]. This network refines features via intrinsic heat-diffusion operators. Such operators are agnostic to the input discretization, thereby extremely robust to varying mesh resolutions and sampling densities. At the same time, it is computationally lightweight. For more details on the choice of backbone, see Appendix A.2.

Pairwise matching The second component \( II \) of our network is a differentiable, multi-scale matching scheme based on the recent pairwise shape matching method DeepShells [21]. The basis of this approach is the energy function

\[ E_{\text{match}}(\mathbf{F}, \mathbf{G}; \tilde{\Pi}) := \sum_{i=1}^{m} \sum_{j=1}^{n} \tilde{\Pi}_{i,j} \| \mathbf{F}_{i} - \mathbf{G}_{j} \|_2^2 \]  

(2)

which has its roots in the theory of optimal transport. For a given transport plan \( \tilde{\Pi} \in [0,1]^{m \times n} \), the energy \( E_{\text{match}} \) specifies the distance between the discrete measures associated with two arbitrary \( l \)-dimensional feature embeddings \( \mathbf{F} = (\mathbf{F}_1, \ldots, \mathbf{F}_m) \in \mathbb{R}^{m \times l} \) and \( \mathbf{G} = (\mathbf{G}_1, \ldots, \mathbf{G}_n) \in \mathbb{R}^{n \times l} \). Taking the minimum over all possible transport plans \( \arg\min_{\tilde{\Pi}} E_{\text{match}}(\mathbf{F}, \mathbf{G}; \tilde{\Pi}) \) results in the Kantorovich formulation of optimal transport [45, 63]. Following the approach described in [21], we obtain a multi-scale shape matching scheme that minimizes Equation (2) in an iterative optimization. For a given pair of shapes \( \mathcal{X}^{(i)} \) and \( \mathcal{X}^{(j)} \), this scheme defines a mapping

\[ \Phi_{\text{match}} : (\mathbf{F}^{(i)}, \mathbf{F}^{(j)}) \mapsto (\Pi^{(i,j)}, \mathbf{V}^{(i,j)}, \ell^{(i,j)}_{\text{match}}). \]  

(3)

We provide further details on the exact update steps of the optimization scheme \( \Phi_{\text{match}} \) in Appendix A.2. From a high-level perspective, \( \Phi_{\text{match}} \) is defined as a deterministic, differentiable function that takes local feature encodings


\[ F^{(i)} \in \mathbb{R}^{m \times l} \text{ and } F^{(j)} \in \mathbb{R}^{n \times l} \text{ as input and predicts a set of correspondences } \Pi^{(i,j)} \in \{0, 1\}^{m \times n}. \] Additionally, \( \Phi_{\text{match}} \) outputs a deformed embedding \( V^{(i,j)} \in \mathbb{R}^{m \times 3} \) of the vertices of \( \mathcal{X}^{(i)} \). These coordinates specify a registered version of the first input shape \( \mathcal{X}^{(i)} \) that closely aligns with the pose of the second input shape \( \mathcal{X}^{(j)} \). The third output \( \ell_{\text{match}}^{(i,j)} > 0 \) is a training loss signal.

### 3.3. Graph-based multi-shape matching

**Shape graph** We now provide details on our multi-shape matching architecture III. To this end, we start by defining an affinity graph

\[ \mathcal{G} := (\mathcal{S}, w), \quad \text{with } w : \mathcal{S} \times \mathcal{S} \to [0, \infty] \] (4)
on the set of training shapes \( \mathcal{S} = \{\mathcal{X}^{(1)}, \ldots, \mathcal{X}^{(N)}\} \), see III in Figure 2 for a visualization. W.l.o.g., we construct \( \mathcal{G} \) as a complete graph (i.e. undirected, fully connected), where a missing edge between \( \mathcal{X}^{(i)} \) and \( \mathcal{X}^{(j)} \) can be specified equivalently by setting the corresponding edge weight to \( w(\mathcal{X}^{(i)}, \mathcal{X}^{(j)}) = \infty \).

We define the pairwise edge weights \( w(\mathcal{X}^{(i)}, \mathcal{X}^{(j)}) \in [0, \infty] \) such that they represent affinity scores between pairs of shapes \( \mathcal{X}^{(i)} \) and \( \mathcal{X}^{(j)} \). By convention, small values \( w(\mathcal{X}^{(i)}, \mathcal{X}^{(j)}) \approx 0 \) reflect that \( \mathcal{X}^{(i)} \) and \( \mathcal{X}^{(j)} \) have a comparably similar geometric structure. Since our method is fully unsupervised, we have no a priori knowledge of such affinities and thereby have to infer them directly from the geometries \( \mathcal{X}^{(i)} \) and \( \mathcal{X}^{(j)} \). To this effect, we propose a simple heuristic for a given pair of shapes \( \mathcal{X}^{(i)} = (V^{(i)}, T^{(i)}) \) and \( \mathcal{X}^{(j)} = (V^{(j)}, T^{(j)}) \) and define the (symmetric) affinity score \( w \) as

\[ w(\mathcal{X}^{(i)}, \mathcal{X}^{(j)}) := \min \left\{ E_{\text{match}}(V^{(i,j)}), V^{(j)}; \Pi^{(i,j)}), E_{\text{match}}(V^{(j,i)}), V^{(i)}; \Pi^{(j,i)}) \right\}. \] (5)

In this context, \( E_{\text{match}} \) is the (self-supervised) matching energy defined in Equation (2), while \( \Pi^{(i,j)} \) and \( V^{(i,j)} \) are the putative correspondences and registrations produced by Equation (3), respectively. The intuition behind this choice of edge weights \( w \) is that a small matching energy \( E_{\text{match}} \) implies a high correspondence accuracy, which is in turn indicative of a high geometric similarity between the input poses \( \mathcal{X}^{(i)} \) and \( \mathcal{X}^{(j)} \).

**Multi-matching** Since we define the edge weights \( w \) according to the self-supervised matching score \( E_{\text{match}} \), small weights \( w(\mathcal{X}^{(i)}, \mathcal{X}^{(j)}) \) generally correlate with a high correspondence accuracy of \( \Pi^{(i,j)} \). Based on this assumption,
we promote cycle-consistency during training via the following expression

\[
\big( i, s_1, \ldots, s_{M-1}, j \big) := \text{Dijkstra} \left( \mathcal{X}^{(i)}, \mathcal{X}^{(j)}; \mathcal{G} \right) \quad (6a)
\]

\[
\Pi_{\text{mult}}^{(i,j)} = \Pi^{(i,s_1)} \circ \Pi^{(s_1,s_2)} \circ \cdots \circ \Pi^{(s_{M-1},j)} \quad (6b)
\]

Rather than matching a pair of shapes \( \mathcal{X}^{(i)} \) and \( \mathcal{X}^{(j)} \) directly, the multi-shape correspondence maps are passed along shortest paths \( \mathcal{X}^{(i)}, \mathcal{X}^{(s_1)}, \ldots, \mathcal{X}^{(s_{M-1})}, \mathcal{X}^{(j)} \) in the graph \( \mathcal{G} \). The approach thereby favors edges with a close affinity, i.e., a small pairwise matching cost \( w(\mathcal{X}^{(s_k)}, \mathcal{X}^{(s_{k+1})}) \). In our experiments, we demonstrate that this simple heuristic yields significant empirical improvements for a broad range of non-rigid matching tasks.

In practice, we utilize the multi-matching from Equation (6) for two distinct use-cases: For once, we can directly query the improved maps \( \Pi^{(i,j)} \) at test time. Additionally, we promote cycle-consistency during training via the following loss term

\[
\ell^{(i,j)}_{\text{cyc}} := E_{\text{match}}(\mathbf{V}^{(i,j)}, \mathbf{V}^{(j)}; \Pi^{(i,j)}_{\text{mult}}). \quad (7)
\]

This loss \( \ell^{(i,j)}_{\text{cyc}} \) imposes a soft penalty on inconsistencies between the registration \( \mathbf{V}^{(i,j)} \) produced by the pairwise matching module \( E_{\text{match}} \), and the multi-shape correspondences \( \Pi^{(i,j)}_{\text{mult}} \) from Equation (6). As before, \( E_{\text{match}} \) is the matching energy defined in Equation (2).

### 3.4. Training protocol

The overall loss function that we minimize during training consists of two individual components

\[
\ell := \mathbb{E}_{\mathcal{X}^{(i)}, \mathcal{X}^{(j)}; \mathbf{S}} \left[ \ell^{(i,j)}_{\text{match}} + \lambda_{\text{cyc}} \ell^{(i,j)}_{\text{cyc}} \right]. \quad (8)
\]

Our complete pipeline is depicted in Figure 2. The whole network is trained end-to-end. In each training iteration, the backbone \( \mathbf{I} \) and pairwise matching module \( \mathbf{II} \) are queried in sequence to produce a pairwise matching for a pair of shapes \( \mathcal{X}^{(i)} \) and \( \mathcal{X}^{(j)} \). The shape graph module \( \mathbf{III} \) then produces the cycle-consistency loss \( \ell^{(i,j)}_{\text{cyc}} \). The shape graph \( \mathcal{G} \) is updated regularly after a fixed number of epochs, taking into account the pairwise matches \( \Pi^{(i,j)} \) for all \( i, j = 1, \ldots, N \). For more details on the training schedule and choices of hyperparameters, see Appendix A.1.

### 4. Experiments

We provide various benchmark evaluations for non-rigid shape matching. We consider classical nearly-isometric datasets in Section 4.1, as well as more specialized benchmarks for matching with topological changes in Section 4.2 and inter-class pairs in Section 4.3. In Section 4.4, we compare different types of shape graph topologies. In Section 4.5, we provide an ablation study of our model, assessing the significance of individual network components.

### Baselines

We compare G-MSM to existing deep learning approaches for unsupervised, deformable 3D shape correspondence. To this end, we consider both standard pairwise matching [20, 21, 24, 51, 56, 57] and multi-matching approaches [10, 22, 26, 28].

### Evaluation

For each experimental setting, we report the mean geodesic correspondence error over all pairs of a given test set category. All evaluations are performed in accordance with the standard Princeton benchmark protocol [31].

### 4.1. Nearly isometric matching

**Datasets** We evaluate our method on four classical, nearly-isometric datasets. FAUST [5], SCAPE [1], SURREAL [61] and SHREC’19 [41]. Following prior work [15, 56, 57], we additionally show generalization results when training on FAUST and testing on SCAPE (F on S), and vice versa. We consider both standard, pairwise baselines [20, 21, 24, 51, 56, 57] and multi-matching approaches [10, 22, 26, 28].

### Table 1. Nearly isometric matching. A quantitative comparison on four nearly-isometric human shape benchmarks, FAUST [5], SCAPE [1], SURREAL [61] and SHREC’19 [41]. Following prior work [15, 56, 57], we additionally show generalization results when training on FAUST and testing on SCAPE (F on S), and vice versa. We consider both standard, pairwise baselines [20, 21, 24, 51, 56, 57] and multi-matching approaches [10, 22, 26, 28].

<table>
<thead>
<tr>
<th>Method</th>
<th>FAUST</th>
<th>SCAPE</th>
<th>F on S</th>
<th>S on F</th>
<th>SUR</th>
<th>SH’19</th>
</tr>
</thead>
<tbody>
<tr>
<td>UFM [24]</td>
<td>5.7</td>
<td>10.0</td>
<td>12.0</td>
<td>9.3</td>
<td>9.2</td>
<td>15.5</td>
</tr>
<tr>
<td>SURFM [51]</td>
<td>7.4</td>
<td>6.1</td>
<td>19.0</td>
<td>23.0</td>
<td>38.9</td>
<td>37.7</td>
</tr>
<tr>
<td>WFM [56]</td>
<td>1.9</td>
<td>4.9</td>
<td>8.0</td>
<td>4.3</td>
<td>38.5</td>
<td>15.0</td>
</tr>
<tr>
<td>DistNet [57]</td>
<td>1.9</td>
<td>2.6</td>
<td>2.7</td>
<td>1.9</td>
<td>8.8</td>
<td>11.0</td>
</tr>
<tr>
<td>DS [21]</td>
<td>1.7</td>
<td>2.5</td>
<td>5.4</td>
<td>2.7</td>
<td>2.7</td>
<td>12.1</td>
</tr>
<tr>
<td>NM [20]</td>
<td>1.5</td>
<td>4.0</td>
<td>6.7</td>
<td>2.0</td>
<td>9.7</td>
<td>2.8</td>
</tr>
<tr>
<td>IsoMuSh [22]</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>3.8</td>
<td>2.5</td>
</tr>
<tr>
<td>CZO [28]</td>
<td>2.2</td>
<td>2.5</td>
<td>–</td>
<td>–</td>
<td>2.2</td>
<td>6.3</td>
</tr>
<tr>
<td>UDM [10]</td>
<td>1.5</td>
<td>2.0</td>
<td>3.2</td>
<td>3.2</td>
<td>3.1</td>
<td>22.8</td>
</tr>
<tr>
<td>SyNoRM [26]</td>
<td>7.9</td>
<td>9.5</td>
<td>21.9</td>
<td>24.6</td>
<td>12.7</td>
<td>7.5</td>
</tr>
<tr>
<td>Ours w/o II</td>
<td>1.7</td>
<td>3.3</td>
<td>4.2</td>
<td>1.7</td>
<td>8.1</td>
<td>6.2</td>
</tr>
<tr>
<td>Ours</td>
<td>1.5</td>
<td>1.8</td>
<td>2.1</td>
<td>1.5</td>
<td>2.1</td>
<td>2.7</td>
</tr>
</tbody>
</table>

22766
**Discussion**  
The results on these four benchmarks are summarized in Table 1. Our method obtains state-of-the-art performance in all considered settings. Remarkably, these results were achieved directly through querying our network, whereas many baselines require correspondence postprocessing [10, 20, 24, 51, 56, 57]. Furthermore, the results underline that the shape graph module \( \text{III} \) plays a critical role in our pipeline for optimal performance.

### 4.2. Matching with topological changes

**Datasets**  
The benchmarks SHREC’Isometry [16] and TOPKIDS [33] focus on matching with topological noise. This is a common phenomenon when working with real scans, where the mesh topology is often corrupted by self-contact of separate parts of the scanned objects. Such topological merging severely affects the correspondence estimation since it distorts the intrinsic shape geometry non-isometrically. The first benchmark SHREC’Isometry [16] contains real scans of different humanoid puppets and hand models. A majority of poses in their ‘heteromorphic’ test set are subject to topological changes, see also Figure 4 for an example. The TOPKIDS [33] dataset contains synthetic shapes of human children where topological merging is emulated by computing the outer hull of intersecting geometries, see Figure 1 for an example.

**Discussion**  
Quantitative results are shown in Figure 3. We observe that topological merging commonly leads to unstable behavior for methods that rely on intrinsic priors like preservation of the Laplace-Beltrami operator [10, 28, 51, 56, 57] or pairwise geodesic distances [24]. It further inhibits approaches that learn to morph input geometries [20, 26] with explicit deformation priors, since merged regions tend to adhere to each other, see, e.g., the discussion on failure cases in [26, Sec. 7]. Our pipeline decreases the correspondence error by a decisive margin of 19% for SHREC’Isometry and 73% for TOPKIDS. We provide a qualitative comparison in Figure 4, as well as additional examples in Appendix D.
While methods with an explicit deformation prior [20, 26] where all nodes are connected to one center node. For a $O$ with sparse connectivity patterns, i.e., TOSCA [8] datasets. SMAL contains inter-class pairs between different animal classes, whereas TOSCA contains nearly-isometric pairs with both animal and human classes.

### 4.4. Sparse graph topologies

Throughout our experiments, we use complete shape graphs $\mathcal{G}$ with a full set of $\frac{N(N-1)}{2}$ edges, as specified in Section 3.3. Here, we explore a few alternative graph topologies with sparse connectivity patterns, i.e., $O(N)$ edges: (ii) Minimal spanning trees (MST), (iii) minimal paths solving the traveling salesman problem (TSP) and (iv) star graphs, where all nodes are connected to one center node. For a detailed discussion and visualizations of these graph types, see Appendix B. We compare (ii)-(iv) to (i) the standard full graph and (v) our pipeline without the graph module III. A given graph type can be employed either for the cycle consistency loss in Equation (7) during training or for map refinement at test time. For a complete picture, we report results for all $5 \times 5 = 25$ possible combinations.

Our results in Table 2 indicate that, while the full graph is generally the most accurate, the sparse topologies often perform comparably, especially MST. This makes them a viable alternative to the full graph in certain scenarios with limited resources, both in terms of the required memory and query time. We provide a comprehensive cost analysis in Appendix C. In most cases, using the shape graph during training is beneficial, even when no graph is available at test time (Table 2, bottom row). This makes it relevant for online applications where not all test pairs are available at once. Regardless of the graph type, it is generally preferable to include some version of our graph module III rather than directly using the pairwise correspondences ‘w/o’.

### 4.5. Ablation study

Our proposed architecture consists of several basic building blocks I-III, as defined in Section 3.2 and Section 3.3. While the shape-graph module III is unique to our approach, the feature backbone I and matching module II can, in principle, be replaced by any analogous off-the-shelf architectures. We compare several popular alternatives: For the feature backbone, we consider I.a the spectral convolution architecture from [21], I.b our network with SHOT [59] input features, I.c the ResNet architecture from [34], I.d PointNet [47], as well as I.e the message passing architecture from [20]. For the differentiable matching module, we compare II.a a functional map layer [44], II.b a single Sinkhorn

<table>
<thead>
<tr>
<th></th>
<th>SH'20 on SMAL</th>
<th>SH'20 on TOSCA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cat</td>
<td>Centaur</td>
</tr>
<tr>
<td>UFM [24]</td>
<td>39.8</td>
<td>32.9</td>
</tr>
<tr>
<td>SURFM [51]</td>
<td>53.4</td>
<td>37.7</td>
</tr>
<tr>
<td>WFM [56]</td>
<td>31.4</td>
<td>20.2</td>
</tr>
<tr>
<td>DiffNet [57]</td>
<td>40.5</td>
<td>18.2</td>
</tr>
<tr>
<td>DS [21]</td>
<td>35.0</td>
<td>10.8</td>
</tr>
<tr>
<td>NM [20]</td>
<td></td>
<td>10.0</td>
</tr>
<tr>
<td>22768</td>
<td></td>
<td>21.7</td>
</tr>
<tr>
<td>UDM [10]</td>
<td>52.6</td>
<td>25.5</td>
</tr>
<tr>
<td>SyNoRiM [26]</td>
<td>10.4</td>
<td>5.7</td>
</tr>
<tr>
<td>Ours w/o III</td>
<td>11.1</td>
<td>3.4</td>
</tr>
<tr>
<td>Ours</td>
<td>10.6</td>
<td>2.6</td>
</tr>
</tbody>
</table>

Figure 5. **Inter-class matching.** (left) A comparison of our approach to the considered baselines on SHREC’20 [17], as well as additional, synthetic test sets from SMAL [65] and TOSCA [8]. (right) Additionally, we visualize the shape graph node embeddings of our approach on TOSCA [8] through 2D multi-dimensional scaling. Since the learned edge weights express affinity scores, shapes with similar geometries tend to cluster together. Shapes with four legs (orange ≙ horse, red ≙ cat, …) and two legs (blue ≙ human, green ≙ gorilla) are linearly separable in the 2D MDS space. Interestingly, the centaur classes’ embedding (purple) lies exactly between these two categories.

**Datasets** The SHREC’20 [17] challenge contains real scans of various four-legged animal models, including: elephant, giraffe, bear, and many more. These geometries were obtained from inhomogeneous acquisition sources, i.e., different types of scanners and 3D reconstruction pipelines. Sparse ground-truth correspondences were obtained through manual annotation. We further assess the generalization to additional test sets from the synthetic SMAL [65] and TOSCA [8] datasets. SMAL contains inter-class pairs between different animal classes, whereas TOSCA contains nearly-isometric pairs with both animal and human classes.

**Discussion** Our approach yields the most stable results overall, see Figure 5. Several baselines suffer from unstable behavior for animals from SHREC’20, because they depend on either noisy SHOT [59] input features [21], intrinsic priors that favor near isometries [28, 56, 57], or both [10, 24, 51]. While methods with an explicit deformation prior [20, 26] perform well on SHREC’20, they do not generalize well to unseen test poses from SMAL and TOSCA. Our method learns a topology-aware shape graph prior and thereby gets the best out of both worlds, i.e., robustness to inter-class pairs and strong generalization to unseen test pairs.
Table 2. Graph topology comparison. We compare the quantitative performance of our model for different graph topologies $G$. Specifically, we revisit the experiment from Figure 3 and report the mean geodesic error on SHREC’Iso [16] and TOPKIDS [33]. The standard ‘full’ graph is compared to three sparse topologies ‘MST’, ‘TSP’, ‘star’ graph, as well as the ‘w/o III’ variant of our pipeline.

<table>
<thead>
<tr>
<th>SH’Iso</th>
<th>Train on</th>
<th>Feature</th>
<th>TOPKIDS</th>
<th>SHREC’20</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Full</td>
<td>MST</td>
<td>TSP</td>
<td>Star</td>
</tr>
<tr>
<td>Test on</td>
<td>5.16</td>
<td>5.09</td>
<td>5.20</td>
<td>5.90</td>
</tr>
<tr>
<td></td>
<td>5.68</td>
<td>5.49</td>
<td>5.50</td>
<td>6.36</td>
</tr>
<tr>
<td></td>
<td>6.08</td>
<td>5.62</td>
<td>5.80</td>
<td>6.94</td>
</tr>
<tr>
<td></td>
<td>5.54</td>
<td>5.26</td>
<td>5.44</td>
<td>6.71</td>
</tr>
<tr>
<td></td>
<td>5.32</td>
<td>5.27</td>
<td>5.42</td>
<td>6.33</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TOPKIDS</th>
<th>Train on</th>
<th>Feature</th>
<th>TOPKIDS</th>
<th>SHREC’20</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Full</td>
<td>MST</td>
<td>TSP</td>
<td>Star</td>
</tr>
<tr>
<td>Test on</td>
<td>7.92</td>
<td>8.13</td>
<td>8.44</td>
<td>11.03</td>
</tr>
<tr>
<td></td>
<td>13.18</td>
<td>12.33</td>
<td>13.10</td>
<td>19.72</td>
</tr>
<tr>
<td></td>
<td>8.61</td>
<td>8.84</td>
<td>8.34</td>
<td>11.92</td>
</tr>
<tr>
<td></td>
<td>10.62</td>
<td>10.64</td>
<td>11.61</td>
<td>13.62</td>
</tr>
</tbody>
</table>

Table 3. Ablation network architecture. We compare several off-the-shelf network architectures for the feature backbone I and matching module II to our full model, as defined in Section 3.2. For each setting, we contrast the results obtained with (✓) and without (✗) the graph-based multi-matching module III.

We then replace either the feature backbone I.a-I.e or matching module II.a-II.c in our method and observe how it affects the accuracy on TOPKIDS [33] and SHREC’20 [17] from Section 4.2 and Section 4.3, respectively. The results are summarized in Table 3. Replacing either module I or II in our approach leads to a drop in performance. Moreover, we see that, regardless of the concrete architecture, our multi-matching approach III (✓ in Table 3) improves the performance over the pairwise matches (✗).
References


[26] Jiahui Huang, Tolga Birdal, Zan Gojcic, Leonidas J Guibas, and Shi-Min Hu. Multiway non-rigid point cloud registration via learned functional map synchronization. IEEE Transac-


