Unsupervised Representation Learning for Diverse Deformable Shape Collections

Sara Hahner, Souhaib Attaiki, Jochen Garcke, Maks Ovsjanikov

Unsupervised Representation Learning for Diverse Deformable Shape Collections

Sara Hahner [⋆,1,2] Souhaib Attaiki [⋆,3] Jochen Garcke [1,2] Maks Ovsjanikov [3]

[1]Fraunhofer SCAI, Sankt Augustin, Germany
[2]Institute for Numerical Simulation, University of Bonn, Germany
[3]LIX, École Polytechnique, Institut Polytechnique de Paris, France

Abstract

We introduce a novel learning-based method for encoding and manipulating 3D surface meshes. Our method is specifically designed to create an interpretable embedding space for deformable shape collections. Unlike previous 3D mesh autoencoders that require meshes to be in a 1-to-1 correspondence, our approach is trained on diverse meshes in an unsupervised manner. Central to our method is a spectral pooling technique that establishes a universal latent space, breaking free from traditional constraints of mesh connectivity and shape categories. The entire process consists of two stages. In the first stage, we employ the functional map paradigm to extract point-to-point (p2p) maps between a collection of shapes in an unsupervised manner. These p2p maps are then utilized to construct a common latent space, which ensures straightforward interpretation and independence from mesh connectivity and shape category. Through extensive experiments, we demonstrate that our method achieves excellent reconstructions and produces more realistic and smoother interpolations than baseline approaches. Our code can be found online: https://github.com/Fraunhofer-SCAI/DISCO-AE/

1. Introduction

Encoding, analyzing, and manipulating 3D surface meshes is a pivotal challenge in 3D computer vision. With the increasing prominence of diverse mesh datasets encompassing humans, animals, and CAD elements, the importance of this issue extends to various applications. These include mesh encoding to lower dimensionality [37], computer-aided engineering [27], and mesh generation [69].

Autoencoders have emerged as a potential solution to this challenge. Standard mesh autoencoders, e.g., [11, 52, 66, 69], begin by calculating vertex-wise features. They then down-sample the mesh using an encoder to compress the shape representation before reconstructing the original mesh with a decoder. Alternate strategies, like [26, 27], implement autoencoding by initially remeshing input meshes to a semi-regular structure. Their autoencoder then handles local patches instead of entire meshes and the added remeshing step often compromises reconstruction quality.

A significant limitation of autoencoders handling meshes is their requirement for meshes in the shape collections to have a 1-to-1 correspondence, meaning all meshes must utilize the same triangulation—a costly and often impractical demand. Moreover, accurately determining correspondences across geometric objects is crucial for numerous computer vision and graphic challenges [9, 15, 48, 68]. Various methods have been developed to address this, with the functional map approach [43] showing particular promise. Both supervised [5, 16, 38, 39, 57] and unsupervised [17, 28, 56] methods have achieved state-of-the-art results in this area. Yet, these shape-matching techniques have not been adapted to mesh autoencoding challenges, necessitating the aforementioned 1-to-1 correspondence.

On the other hand, creating a unified and interpretable embedding space for meshes poses another challenge. Techniques that down-sample the input mesh, lead to an embedding space dependent on mesh connectivity. Others employ mean or max pooling for vertex features to generate a global shape feature, but this may not always create smooth embedding manifolds.

In our study, we address the issues mentioned above with a novel mesh autoencoder, trained in an entirely unsupervised manner, that forms a universal latent space unaffected by the shape type or mesh connectivity, enhancing interpretability. For this, we introduce a spectral pooling method to establish this shared space, relying on point-to-point maps between shapes. Advocating for unsupervised methods, we utilize the functional maps pipeline [43] to extract these maps, allowing us to define an embedding

(⋆) denotes equal contribution

space that transcends mesh connectivity and shape categories. The generated shape features reside on a smooth manifold, facilitating interpretable sampling for mesh generation.

Overall, our primary contributions are:

- The introduction of a spectral pooling method that disregards mesh connectivity, yielding a shared embedding space for diverse mesh types and categories.
- A pioneering unsupervised training technique to obtain a mesh autoencoder independent of a fixed mesh template.
- Demonstrations showcasing our method's capacity to reconstruct superior-quality meshes and generate an interpretable embedding space optimal for shape sampling and manipulation.

2. Related Work

In this section, we review previous works related to our research. We organize them into three main categories.

Mesh Autoencoders and Generative Models Deformable shape representation and generation is a well-studied domain [8, 13, 23, 34]. [37] and [52] (CoMA) introduce some of the first mesh autoencoders. The authors of CoMA, the Neural3DMM network [11], and [66] utilize mesh downsampling and mesh upsampling layers for pooling and unpooling, which are combined with either spectral or spiral convolutional layers. By manually choosing latent vertices for the embedding space, [69] defines a MeshConv autoencoder that allows interpolating in the latent space. All the above-mentioned mesh convolutional autoencoders work only for collections of meshes with the same connectivity because the pooling and/or convolutional layers depend on the adjacency matrix. [26] (spatial CoSMA) introduce a patch-based approach. The meshes have to be remeshed to semi-regular mesh connectivity. The resulting regular patches are input separately to an autoencoder using spatial convolution, allowing for an analysis of meshes of different sizes. Spectral CoSMA [27] combines this patch-based approach with Chebyshev convolutions [14] on the patches. The MeshCNN architecture [29] can be implemented as an encoder and decoder. Nevertheless, the pooling is feature-dependent, so the embeddings can be of different significance.

For surfaces that are represented as signed distance functions and in other implicit representations, [22] and [46] achieve good results in shape reconstruction and completion. Nevertheless, their generalization and scalability are often limited to a small set of deformations and require big training data. Another parallel line of work is representation learning on point clouds [1, 51, 67]. In theory, these methods can handle surface meshes when disregarding the faces defining the surface mesh. However, these methods only reconstruct and generate point clouds, which is a different and more straightforward task compared to what our work aims for because of their permutation invariance.

The compact representation of the input data by the autoencoder can be used for data generation and manipulation. The features are randomly sampled or combined linearly, generating shapes in positions that the user controls. [19, 27, 52] show mesh generative results by sampling from an autoencoder's or variational autoencoder's mesh feature space. Other generative approaches [60, 64] rely on a non-learned deformation representation of meshes of fixed connectivity.

Shape Matching Shape matching has been extensively studied in computer graphics. While a comprehensive review is beyond the scope of this paper, interested readers can refer to recent surveys [12, 24, 25] for a more in-depth discussion. One of the methods most related to our work is the functional map pipeline, which was introduced in [43, 44] and has since been extended in many follow-up works [2, 3, 5, 6, 18, 40, 53]. The main advantage of this method is that it transforms the problem of optimizing for a point-to-point map (which is quadratic in the number of vertices) into the optimization of a functional map (which consists of small quadratic matrices), making the optimization process feasible. To find the functional map, earlier works relied on hand-crafted feature functions defined on source and target shapes, such as HKS [59], WKS [7], or SHOT [55] features. Follow-up research improved the pipeline by introducing additional regularization [33, 42], and proposing efficient refinement methods [53]. More recently, the functional map pipeline has been incorporated into deep learning, with the seminal work of [36] and subsequent works [5, 16, 35, 56] using differentiable functional map losses and regularization to learn feature functions with neural networks. Another line of work focused on making the learning unsupervised [2, 4, 17, 54], which can be useful in the absence of ground truth correspondences. This was achieved by imposing structural properties such as bijectivity and orthonormality on functional maps in the reduced basis [54], penalizing the geodesic distortion of the predicted maps [28], or combining intrinsic and extrinsic shape alignment [17]. However, all of these works focused on establishing correspondences and did not investigate any relationship with shape reconstruction or generation.

Structure of the Feature Space The representation learned by an autoencoder typically resides in a lower-dimensional representation space than the input. In this work, our goal is to create a representation space that is shared among different mesh representations and collections. A common method for point clouds is performing (weighted) vertex-wise feature averaging [49, 50]. When neglecting the surface structure defined by the faces, one can apply this approach to the vertices only. However, this

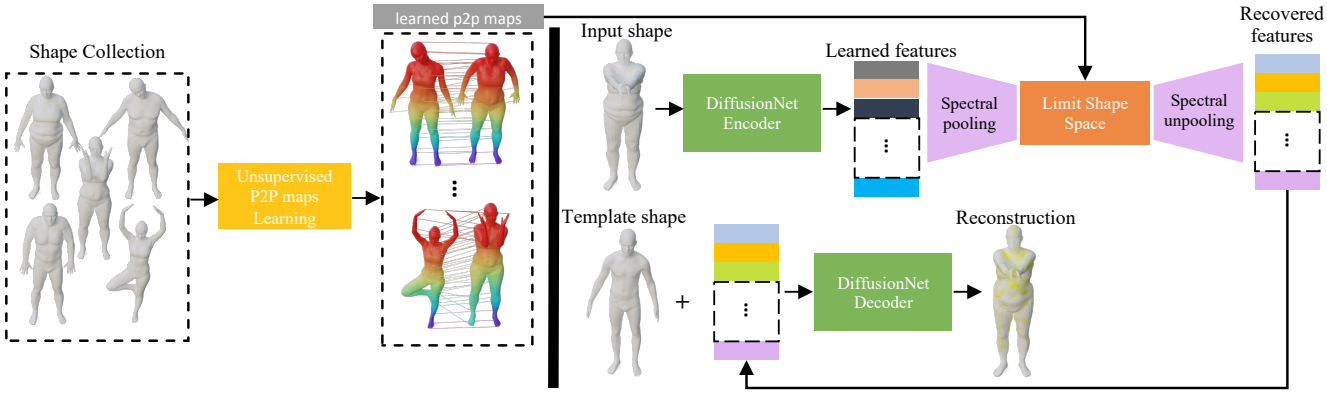

Figure 1. **Method overview** Our method consists of two stages. In the first stage (left), we train a deep functional map network to extract p2p maps between the input shapes (see Section 4.2). These p2p maps are then used for spectral pooling by constructing the limit shape space, which is used as the embedding space in our mesh autoencoder in stage 2 (right, see Sections 4.1 and 4.3).

approach is highly sensitive to the distribution of vertices in 3D space, and it cannot guarantee that features of different shapes lie in the same manifold a priori. An alternative approach is to use a template and analyze the features with respect to the template [20, 21, 32]. However, the use of a template can introduce bias. To avoid this, some methods construct a new 3D template shape that resembles the centroid of the collection [31]. In our work, we avoid constructing such a shape by using the limit shape basis CCLB [30]. This approach defines a latent shape in the spectral space to which all shapes are connected via a functional map, thereby avoiding embedding it in the ambient space and introducing potential bias.

3. Motivation, Notation & Background

In this section, we express our motivation for creating an autoencoder that can process meshes with varying connectivities. This is a shift from traditional autoencoders, which mainly work with point clouds and meshes with fixed connectivities. We also touch upon the functional map framework and the idea of limit shape construction. These concepts are crucial to our proposed method. We use the same notation throughout to make the paper easier to follow.

3.1. Motivation

Autoencoders have made significant strides in learning compact representations across various data types. In the 3D domain, they have been particularly successful with point cloud data due to its permutation-invariance property, streamlining the encoding and decoding processes [1, 45, 63, 65]. However, when applied to triangular meshes, this strength becomes a limitation.

Triangular meshes, in contrast to point clouds, encapsulate the detailed geometry and topology of 3D surfaces, essential for applications like computer graphics where accurate 3D representations underpin realistic renderings. They impose an inherent structure on the 3D data, encoding both geometric and topological relationships among vertices. Unlike the permutation-invariant nature of point clouds, the order in triangular meshes is pivotal as it defines the mesh's connectivity. Tampering with this order could obliterate connectivity data, thereby diminishing the mesh's representational utility. This distinctive characteristic of meshes makes tailoring autoencoders for them notably challenging.

The prevailing approaches to address this challenge often assume that all meshes maintain a 1-to-1 correspondence, meaning they possess identical mesh connectivity [11, 52, 69]. While this perspective facilitates preserving mesh structures during encoding and decoding through mesh resampling, it also restricts the versatility of these methods. In practice, a strict 1-to-1 correspondence is an exception rather than the rule. Forcing diverse meshes into identical connectivity introduces intricate challenges, often necessitating remeshing and manual fine-tuning. Such remeshing might produce distortions, undermining the original mesh's quality. Furthermore, when the mesh structure encapsulates salient features about an object, remeshing might not be just unfeasible but also undesirable.

Another ambition in the field is to situate the meshes within a shared embedding space, allowing for both comparative and manipulative operations on the shapes. Contemporary mesh autoencoders, however, hinge on fixed mesh connectivity to form this shared space [11, 26, 27, 52].

Motivated by these challenges, our work seeks to develop a novel Mesh Autoencoder for **DI**verse **S**hape **CO**llections (DISCO-AE) capable of handling arbitrary triangular meshes, thereby eliminating the need for 1-to-1 correspondence, and representing them in a joined embedding space.

3.2. Notation

We consider a 3D shape S_i, represented as a triangular mesh comprising n_i vertices. We obtain its cotangent Laplace-

Beltrami decomposition [61] and represent the first k eigenvectors of S_i in the matrix $\Phi_i \in \mathbb{R}^{n_i \times k}$. Additionally, we construct a diagonal matrix $\Delta_i \in \mathbb{R}^{k \times k}$, with its diagonal elements containing the first k eigenvalues of S_i. We also define the diagonal matrix of area weights as $M_i \in R^{n \times n}$. It should be noted that Φ_i is orthogonal with respect to M_i and that $\Phi_i^\top M_i \Phi_i = I_k$, where I_k denotes the $\mathbb{R}^{k \times k}$ identity matrix. We further denote $\Phi_i^\dagger = \Phi_i^\top M_i$ and use the (left) Moore-Penrose pseudo-inverse symbol, $\cdot^\dagger$, to represent it.

3.3. Functional map pipeline

We use the notation S_1 and S_2 to refer to a source and target shape, respectively. The pointwise map $T_{12} : S_1 \rightarrow S_2$ is defined as the function that maps each vertex in S_1 to a corresponding vertex in S_2. To represent this map, we use the matrix $\Pi_{12} \in \mathbb{R}^{n_1 \times n_2}$, which takes the value 1 if $T_{12}(i) = j$, and 0 otherwise. However, with an increasing number of vertices in the shapes, the size of the matrix Π_{12} grows quadratically, which is computationally infeasible.

To address this issue, we adopt the functional map paradigm proposed in [43]. This approach reduces the dimensionality of Π_{12} by representing it in the spectral basis. Specifically, we construct the functional map C_{21}, which maps functions defined on S_2 to functions defined on S_1, using the expression $C_{21} = \Phi_1^\dagger \Pi_{12} \Phi_2$. The functional map has a small size of $(k \times k)$, with k usually around 30, making the optimization process feasible.

To find the functional maps that map S_1 and S_2, we first obtain two d-dimensional feature functions, also known as probes, F_1 and F_2 defined on S_1 and S_2 respectively ($F_i \in \mathbb{R}^{n_i \times d}$). We then compute the coefficients $\mathbf{A}_i$ of the feature functions in their corresponding reduced basis using $\mathbf{A}_i = \Phi_i^\dagger F_i$. Next, we formulate an optimization problem:

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

where $\mathbf{C}$ is the sought-after functional map.

3.4. Canonical Consistent Latent Basis

Given a collection of related 3D shapes $S_1, \ldots, S_n$, and a set of functional maps between some shape pairs, we build a functional map network on the collection as follows. We construct a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the i-th vertex represents the functional space of the shape S_i, and the edge (i, j) exists if the functional maps C_{ij} and C_{ji} are given, in which case, the graph is symmetric. We assume that our graph is connected, which means that there exists a path between any two shapes in the collection.

With this construction in hand, we can translate functions between any shapes S_i and S_j in the shape collection. Nevertheless, we do not have a common basis. We solve this by using the limit shape construction as in [62], which provides a latent basis Y_i for the collection's shape features, such that $C_{i,j} Y_i \approx Y_j, \forall i, j$. These latent bases $Y_i \in \mathbb{R}^{k_1 \times k_1}$ (k_1 is the same dimension of the functional maps) can be interpreted as functional maps from a *latent shape* to each shape S_i.

To further enhance the stability of this construction and eliminate shape metric ambiguity, [30] introduced the canonical consistent latent basis (CCLB) $\widetilde{Y}_i \in \mathbb{R}^{k_1 \times k_2}$, which has been shown to yield better results. The CCLB enables unbiased comparisons of the shape features in the collection. Therefore, we use this common basis to define the embedding space of our autoencoder, which captures the diversity of our shape collection. To reduce the instabilities that could rise from imperfect p2p maps, we chose a star graph topology for $\mathcal{G}$ by connecting all shapes to the corresponding template shape.

4. Method

In this section, we introduce our proposed model for shape representation and generation, resolving the challenges motivated in the previous section. For that, we introduce a novel spectral mesh pooling and present an unsupervised learning method of functional maps to construct point-to-point maps between a collection of shapes, This is the first stage of our approach, for which we provide an overview in Figure 1. The second stage of our model is an autoencoder making use of the novel spectral pooling. We refer to the autoencoder for **DI**verse 3D **S**hape **CO**llections as **DISCO-AE**.

4.1. Spectral Pooling

We develop a spectral mesh pooling operator to reduce the dimensionality of the meshes in the spectral domain to handle meshes of different connectivity and represent them in a joined low-dimensional embedding space.

In the case of classical representation learning for 2D images with convolutional networks, all image samples have a fixed size and are in 1-to-1 correspondence. The convolutional filters calculate vertex-wise features, then pooling summarizes many vertex-wise features, reducing the number of pixels. This is done uniformly for all the images in correspondence, and hence, features from different samples are comparable to each other. Therefore, pooling in 2D can also be interpreted as a projection from a high dimensional basis to a lower dimensional basis functions. Here, the cardinality of the basis is equal to the number of pixels. Because of the 1-to-1 correspondence, all the images are described on the same basis. A similar pooling operator cannot be constructed for meshes with different mesh connectivities. We can only obtain point-to-point maps between the shapes that allow the projection of a function from one shape to another.

To solve the pooling for meshes, we propose to adapt the CCLB method (initially developed for deformation detec-

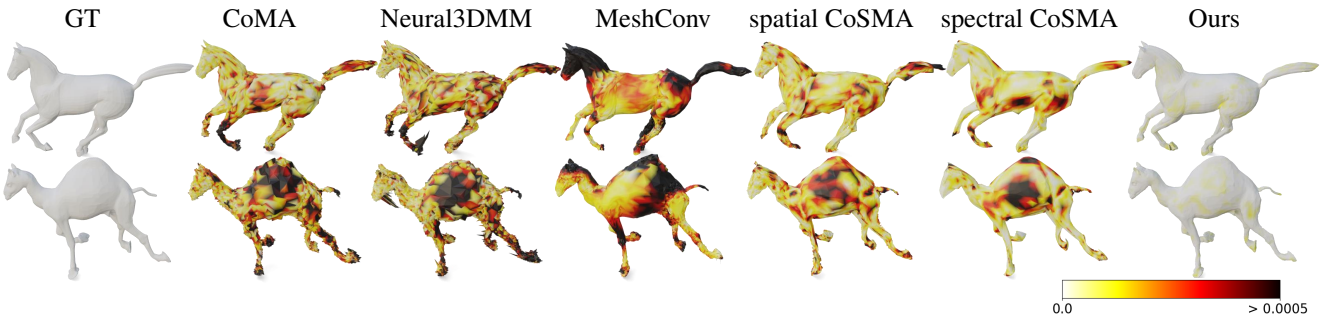

Figure 2. Reconstructed meshes from the *GALLOP* dataset. Vertex-wise error is highlighted.

tion) and introduce a novel intrinsic spectral mesh pooling. We project vertex-wise features that are calculated for every shape separately to the common CCLB basis, reducing the dimension from the number of vertices to the size of the limit shape. We calculate the limit shape basis CCLB as described in 3.4. It has dimension k_2 and uses eigendecompositions of the Laplacians of size $k_1 \geq k_2$. It will be the common basis for the low-dimensional embedding space. For the spectral unpooling, we project the features from the limit shape basis back to the vertex representation.

Note that in the special case, when the dimension of the limit shape equals one ($k_1 = k_2 = 1$), the spectral pooling corresponds to a global $\pm$ mean pooling for all the shapes in the collection. The spectral unpooling duplicates the average feature into the vertices of the shape. We formally state and prove this observation in the supplementary materials.

4.2. Unsupervised Maps Extraction

This step aims to generate point-to-point (p2p) maps between a collection of shapes for training the autoencoder. While most mesh autoencoder requires p2p supervision, obtaining p2p maps is challenging since it requires significant labeling effort, which is prohibitive. To overcome this challenge, we propose learning approximate p2p maps in an unsupervised manner and introducing additional regularization in the loss to rectify the defaults in the maps.

To achieve this, we learn the maps using the unsupervised functional maps setting. We followed the approach of [16] by training a DiffusionNet network [57] to generate feature functions that will estimate a functional map between a source and a target shape. We supervise the training by imposing structural properties on the functional maps.

Specifically, given a source and target shape S_1 and S_2, we first extract d-dimensional feature functions F_1 and F_2, respectively using DiffusionNet. We then project these features onto the reduced Laplacian basis $\mathbf{A}_i = \Phi_i^\dagger F_i$. Next, we estimate the functional map between S_1 and S_2 using:

$$\arg\min_{\mathbf{C}} \|\mathbf{C}\mathbf{A}_1 - \mathbf{A}_2\|_F^2 + \lambda\|\mathbf{C}\Delta_1 - \Delta_2\mathbf{C}\|_F^2. \quad (2)$$

The second term is a regularization that promotes the isometry of the maps, as described in [43]. This operation is differentiable. To train the network, we predict the functional map in both directions (*i.e.*, $\mathbf{C}_{12}$ and $\mathbf{C}_{21}$) and then penalize the deviation of the predicted maps from bijectivity and orthogonality. The first loss requires the maps to be the inverse of each other, while the second loss regularizes the maps to be locally area-preserving, as several previous works demonstrate [54, 56]. We can write these losses as:

$$L = \|\mathbf{C}_{12}\mathbf{C}_{21} - I\|_F^2 + \sum_{i,j\in\{1,2\}} \|\mathbf{C}_{ij}^\top\mathbf{C}_{ij} - I\|_F^2 \quad (3)$$

Once the network is trained, we extract functional maps between all pairs of shapes and convert them to p2p maps. To improve the quality of the maps, we use the recent refinement method ZoomOut [40]. This method navigates between the spectral and spatial domains while progressively increasing the number of spectral basis functions. The final maps are then used to train the autoencoder.

4.3. Our Architecture

Given a shape collection of meshes that can have different connectivity, we define our architecture employing the contributions explained in the previous paragraphs. Using existing (ground truth) or unsupervised learned point-to-point maps (as in Section 4.2), we calculate functional maps between the shapes and then construct the functional map network, as well as the limit shape basis CCLB for the introduced spectral pooling. In addition, we chose a set of template meshes from the collection for the different categories of meshes, which will be used for the reconstructions.

Our autoencoder makes use of the surface-based convolutional network DiffusionNet [57], which has proven to learn discretization agnostic vertex-wise shape features. We input the vertex 3D coordinates of shape S_i to the **encoder**. Four trainable DiffusionNet Blocks [57] are applied to calculate F-dimensional vertex-wise features. Then we apply spectral pooling, and these features are projected to the CCLB by multiplying them from the left by $Y_i^\dagger\Phi_i^\dagger$. This low-dimensional representation z_i of dimensionality $F \cdot k_2$ is now independent of the mesh connectivity of S_i because it is represented in the common CCLB basis.

The **decoder** applies spectral unpooling and projects the features represented in the CCLB to the template shape S_t

by multiplying it by $\Phi_t Y_t$ from the left. At this point, we concatenate the vertex-wise 3D coordinates of the template shape to the projected features to provide more information for the reconstruction of the input shape. Finally, four trainable DiffusionNet Blocks reconstruct the 3D coordinates of the input shape on the template mesh's vertices.

4.4. Losses

Our autoencoder is fully differentiable, and we denote the input shape as S. The encoder and the decoder are respectively represented as enc and dec, the reconstruction is $X = dec(enc(S))$. We train our network using two losses.

Point-to-point (p2p) loss: Given a point-to-point map Π (either ground truth or extracted by the first stage) between the template and the input shape, the p2p loss is defined as $L_1 = \|\Pi S - X\|_F^2$. However, in the case of unsupervised maps, this loss may provide inaccurate signals as the p2p map is often faulty and not entirely correct. To address this issue, we use an additional loss.

Reconstruction loss: Given the reconstruction X, we construct the matrix D^X such that $D^X_{i,j} = \|X_i - X_j\|_F^2$. We create the matrix D^S for ΠS in the same manner. The reconstruction loss is $L_2 = \|D^S - D^X\|_F^2$. This loss computes the cumulative reconstruction error and each point receives reconstruction feedback from the other $n-1$ points. Thus, even if the p2p map is faulty in some places, the faulty points receive signals from the non-faulty ones. As this loss is rotation invariant, it cannot be used alone. Our final loss combines the two losses: $L = L_1 + \lambda L_2$.

5. Experiments

We evaluate our architecture on various tasks using three different shape collections. We denote our method as **DISCO-AE** for DIverse Shape COllection Auto-Encoder.

5.1. Shape Collections

We conduct experiments using three distinct datasets previously utilized in recent studies [26, 27].

The *GALLOP* shape collection contains triangular meshes representing a motion sequence with 48 timesteps from a galloping horse, elephant, and camel [58]. The galloping movement is similar but the meshes representing the surfaces of the three animals differ in connectivity and the number of vertices. We use the last 14 timesteps for testing.

The *FAUST* collection contains 100 meshes [9]. The irregular surface meshes represent 10 different bodies in 10 different poses. We apply two different train-test splits, following previous works [27]. In the first setting, known as "unknown poses", the network is trained on 8 poses out of 10, and tested on the remaining 2, while in the second setting, known as "unknown individuals", the network is trained on 8 individuals and tested on the remaining 2.

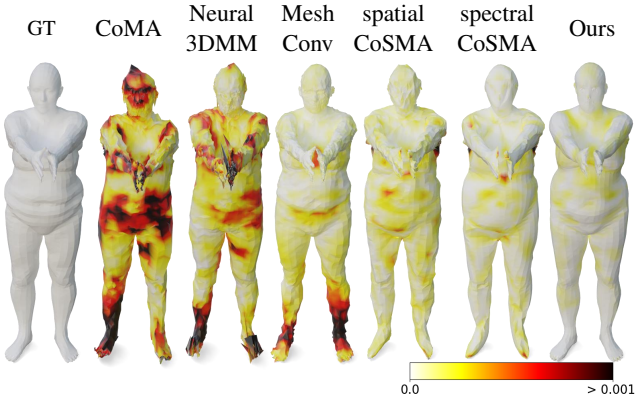

Figure 3. Reconstructed meshes from the *FAUST* dataset of the "unknown individuals" setup. Vertex-wise error is highlighted.

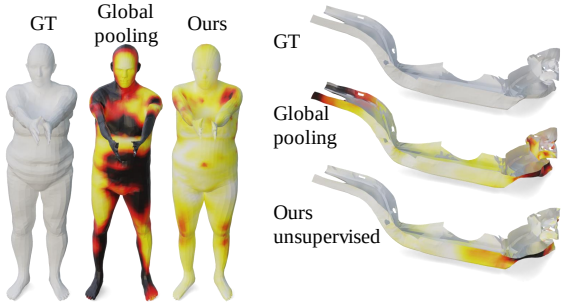

Figure 4. Reconstructions from the unsupervised experiments on the *TRUCK* and *FAUST* datasets. Vertex-wise error is highlighted using the same color range as for the supervised experiments.

The *TRUCK* shape collection [27] contains 32 completed frontal car crash simulations of 6 different components [41]. Only 10 simulations are included in the training set. In this dataset, the components represented by surface meshes often deform in different patterns during the crash. One goal is to detect clusters corresponding to different deformation patterns in the components' embeddings in order to speed up the analysis of car crash simulations [10].

5.2. Results

We compare our method to five recent baseline architectures: CoMA [52], Neural3DMM [11], MeshConv [69], spatial CoSMA [26], and spectral CoSMA [27]. For all baseline autoencoders, we chose embedding sizes following previous works. The first three do not allow an analysis of meshes with different mesh connectivity by the same trained architecture. The latter two methods allow an analysis of different meshes with different connectivity after being remeshed to a semi-regular mesh representation by inputting patches of the meshes to the AE. Nevertheless, their shape features depend on the semi-regular mesh connectivity; hence, the embedding space is not joint. The reconstructed semi-regular meshes are projected back to the original meshes using a parametrization to calculate the error. All these baseline mesh AE are supervised, so we compare

Method	Unsupervised	No remesh	*FAUST* Unknown poses	*FAUST* Unknown indiv.	*GALLOP* Camel	*GALLOP* Elephant	*GALLOP* Horse	*TRUCK* (×100)
CoMA	✗	✓	569.3 ± 203.1	28.3 ± 6.4	7.8 ± 1.4	24.3 ± 4.4	3.2 ± 0.3	-
Neural3DMM	✗	✓	246.2 ± 5.4	10.4 ± 0.9	12.4 ± 0.1	29.7 ± 3.5	4.7 ± 0.1	-
MeshConv	✗	✓	18.2 ± 2.2	3.5 ± 0.4	9.2 ± 0.4	- [1]	7.3 ± 0.1	-
spatial CoSMA	✗	✗	2.8 ± 0.2	1.3 ± 0.1	4.2 ± 0.03	17.8 ± 0.8	1.8 ± 0.02	187.8 ± 11.7
spectral CoSMA	✗	✗	**1.1 ± 0.01**	0.9 ± 0.02	3.3 ± 0.01	20.0 ± 0.3	1.2 ± 0.01	15.7 ± 0.5
DISCO-AE - sup. (ours)	✗	✓	2.4 ± 0.08	**0.7 ± 0.01**	**0.3 ± 0.03**	**1.09 ± 0.08**	**0.11 ± 0.01**	**1.01 ± 0.1**
Global pooling	✓	✓	475 ± 26.2	35.5 ± 1.0	25.2 ± 5.6	22.4 ± 0.2	3.9 ± 0.9	14.6 ± 1.9
DISCO-AE - unsup. (ours)	✓	✓	**4.3 ± 0.1**	**2.0 ± 0.05**	**6.8 ± 0.3**	**21.0 ± 0.2**	**1.2 ± 0.02**	**10.9 ± 0.6**

Table 1. Euclidean errors between the reconstructed and original mesh of the *FAUST*, *GALLOP*, and *TRUCK* datasets. The reported numbers are mean errors over 3 runs randomly initialized. ± denotes the standard deviation.

our approach to them using supervised point-to-point maps.

The second is to train the autoencoder using the unsupervised maps produced by the first stage, see section 4.2. As a comparison, we construct a baseline method that uses unsupervised point-to-point maps and global average pooling instead of the introduced spectral mesh pooling. This corresponds to the case when the dimensionality of the CCLB is 1 ($k_1 = k_2 = 1$), see section 4.1.

5.2.1 Mesh Reconstructions

We initiate our analysis by conducting a conventional reconstruction experiment. First, we encode a shape S from the test set, which was never seen during the training phase, into a latent code. This is decoded subsequently using our decoder. We compare the output to the initial shape to assess the reconstruction. We sum up the vertex-wise squared Euclidean distances between the vertex coordinates of the input shapes and their reconstructions to determine the reconstruction error. To obtain uniform results, we normalize all meshes into the range $[-1, 1]$. We report all reconstruction errors in Table 1.

For the *FAUST* dataset, our supervised method achieves the best result in the "unknown individuals" setting and the second-best result in the "unknown poses" setting (see Figure 3). In addition, our results are more stable than some of the baselines, as indicated by the standard deviation. Our unsupervised results are better than the supervised results that do not require any remeshing, which demonstrates the usefulness of our approach and the regularization introduced by the losses to mitigate errors in the maps. Additionally, our spectral pooling strongly improves the reconstruction quality for the unsupervised experiments compared to using global pooling in the encoder, see Figure 4.

For the *GALLOP* dataset, we train our network on all categories in the supervised setting. However, due to the highly non-isometric nature of the three categories, most unsupervised methods for shape matching fail. Thus, we train our unsupervised method on each category individually. The mesh-dependent baselines are also trained on each animal separately. Only spatial and spectral CoSMA train on the three animals together since mesh patches are input separately. Our supervised method achieves the best results for all categories, see Table 1. Reconstructed meshes are visualized in Figure 2. Concerning our unsupervised method, it achieves comparable results with the baselines and outperforms the unsupervised global pooling approach. This demonstrates that learning high-quality mesh autoencoders is possible even in the absence of ground truth maps.

Finally, we report in Table 1 the result on the *TRUCK* dataset. Due to its big size, we only test our method against the best two performing methods. Once again, our method achieves the best results in the supervised case. Additionally, our unsupervised reconstruction quality is superior to all supervised baselines. We provide visualizations of the reconstructed meshes in the supplementary material for the supervised and Figure 4 for the unsupervised methods.

Qualitatively, our reconstructed meshes are smooth, deform naturally, and do not have any outlier vertices, which is not the case for some baseline methods. The provided reconstructed meshes from all three datasets and supervised and unsupervised experiments in Figures 2 to 4 are smooth and have the lowest reconstruction error.

5.2.2 Low-Dimensional Embeddings

For every mesh from the collections, we obtain a hidden representation of size $k_2 \times F$. The shape features from the same collection can be visualized in 2D or 3D using a principal component analysis [47], see Figure 5.

Similar to the other approaches, we embed the different shape categories separately from each other. In the case of the *FAUST* dataset, several clusters form in the embedding space of the unsupervised experiment, which corresponds to different positions. Additionally, along the horizontal axis, the position of the arms can be split into raised or not raised.

Additionally, for the *first time*, we can jointly visualize the features from various shapes of different connectivity in a common basis. It allows for a joint visualization of the galloping sequences of camel, horse, and elephant from

[1] MeshConv AE for the elephant is too large to train on 40 GB GPU.

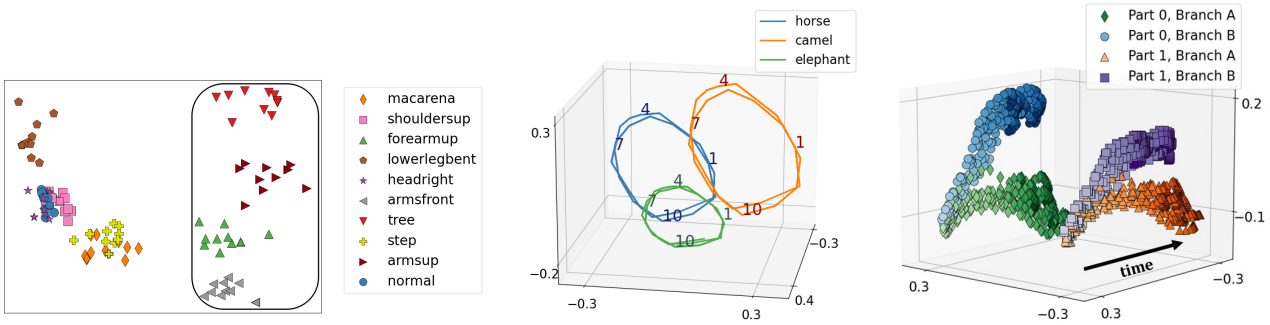

Figure 5. Embeddings in 3D or 2D of the learned representations in the common basis. Left: *FAUST* positions marked with a triangle raise the arms. Middle: galloping sequences from the *GALLOP* dataset with timesteps provided in the plot. Right: two *TRUCK* components deform in two clusters over time, corresponding to the deformation patterns (Branch A and B) visualized in the supplementary material.

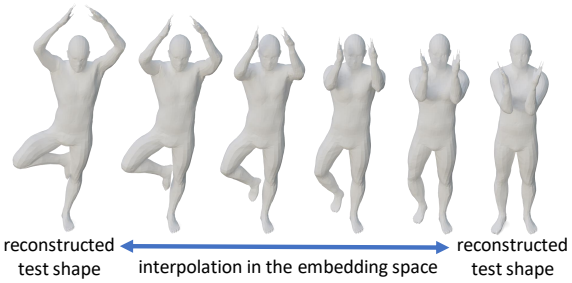

Figure 6. Interpolating between different *FAUST* test shapes.

the *GALLOP* shape collection. The CoSMA baselines, on the other hand, only generate embeddings of every animal separately. Figure 5 visualizes the learned features in 3D for the supervised experiment because the unsupervised one was conducted on the animals separately. The sequences align over time up to translation but are still separated from each other, which captures the different shape categories.

For two *TRUCK* components, we aim to detect two clusters corresponding to a different deformation behavior, similarly to [27]. These different *TRUCK* components can, for the *first time*, be visualized together using the representation in the CCLB. The two deformation branches in two different components are split along the same axis of the 3-dimensional embedding space, and the features of both components align over time, see Figure 5 for the embedding from the unsupervised experiment. This visualizes nicely that the deformation of the two components manifests in similar deformation patterns.

5.2.3 Shape Generation and Manipulation

To show that the shape features lie on a smooth manifold and that the network is not overfitting to the training samples, we generate new shapes by sampling from the latent feature space from the supervised "unknown individuals" setting. We conduct three different generative experiments on the *FAUST* shape collection: interpolation of two test shapes (Figure 6), as well as generation of combined positions and feature transfer between two different bodies (supplementary material). The figures show the smooth and well-formed generated shapes with correctly and naturally positioned limbs. While the feature transfer results can be compared to an actual shape from the collection, our interpolation and position combination experiments create well-formed samples that cannot be found in the shape collection. Additionally, the combination of positions and feature transfer shows that our embedding space allows algebraic manipulation (addition and subtraction) of shape embeddings.

6. Conclusion & Limitations

In this work, we introduce a novel unsupervised method for learning representations of diverse deformable shape collections. Our presented autoencoder architecture reconstructs shapes in higher quality than various baseline methods. Additionally, the computed features of meshes with different connectivity, non-isometric deformations and from different categories lie in the same embedding space. This smooth embedding space, which allows for interpolation and algebraic manipulation, motivates the application of spectral pooling for generative models.

One limitation of our work is that it does not yet handle shape collections with high non-isometry, such as the *GALLOP* shape collection, where we were unable to learn good point-to-point maps between different classes (*i.e.*, between horses and elephants). While our network uses a set of fixed templates for reconstruction, it would be interesting to investigate whether the decoder can generate multiple mesh topologies without the use of a template. We leave this as future work.

Acknowledgements We gratefully thank the reviewers for their valuable suggestions. Parts of this work were supported by the ERC Starting Grant No. 758800 (EXPROTEA), the ANR AI Chair AIGRETTE, and the GlobalMath-Network from the Hausdorff Center for Mathematics.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. *35th International Conference on Machine Learning, ICML 2018*, 1:67–85, 2018. 2, 3

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

[3] Souhaib Attaiki and Maks Ovsjanikov. Understanding and improving features learned in deep functional maps. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[4] Souhaib Attaiki and Maks Ovsjanikov. Shape non-rigid kinematics SNK: A zero-shot method for non-rigid shape matching via unsupervised functional map regularized reconstruction. In *Advances in Neural Information Processing Systems*, 2023. 2

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

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

[7] 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. 2

[8] Tristan Aumentado-Armstrong, Stavros Tsogkas, Allan Jepson, and Sven Dickinson. Geometric disentanglement for generative latent shape models, 2019. 2

[9] Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. Faust: Dataset and evaluation for 3d mesh registration. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3794–3801, 2014. 1, 6

[10] Bastian Bohn, Jochen Garcke, Rodrigo Iza-Teran, Alexander Paprotny, Benjamin Peherstorfer, Ulf Schepsmeier, and Clemens August Thole. Analysis of car crash simulation data with nonlinear machine learning methods. *Procedia Computer Science*, 18:621–630, 2013. 6

[11] Giorgos Bouritsas, Sergiy Bokhnyak, Stylianos Ploumpis, Stefanos Zafeiriou, and Michael Bronstein. Neural 3d morphable models: Spiral convolutional networks for 3d shape representation learning and generation. *Proceedings of the IEEE International Conference on Computer Vision*, 2019-Octob:7212–7221, 2019. 1, 2, 3, 6

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

[13] Luca Di Cosmo, Antonio Norelli, Oshri Halimi, Ron Kimmel, and Emanuele Rodolà. Limp: Learning latent shape representations with metric preservation priors. In *European Conference on Computer Vision*, 2020. 2

[14] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016. 2

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

[16] 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, 5

[17] Marvin Eisenberger, Aysim Toker, Laura Leal-Taixé, and Daniel Cremers. Deep shells: Unsupervised shape correspondence with optimal transport. In *Advances in Neural Information Processing Systems*, pages 10491–10502. Curran Associates, Inc., 2020. 1, 2

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

[19] Simone Foti, Bongjin Koo, Danail Stoyanov, and Matthew J Clarkson. 3d shape variational autoencoder latent disentanglement via mini-batch feature swapping for bodies and faces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18730–18739, 2022. 2

[20] Vignesh Ganapathi-Subramanian, Olga Diamanti, Soeren Pirk, Chengcheng Tang, Matthias Niessner, and Leonidas Guibas. Parsing geometry using structure-aware shape templates. In *2018 International Conference on 3D Vision (3DV)*, pages 672–681, 2018. 3

[21] Jochen Garcke, Sara Hahner, and Rodrigo Iza-Teran. Alignment of highly resolved time-dependent experimental and simulated crash test data. *International Journal of Crashworthiness*, 0(0):1–15, 2022. 3

[22] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *International Conference on Machine Learning*, pages 3789–3799. PMLR, 2020. 2

[23] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry. 3d-coded : 3d correspondences by deep deformation. In *ECCV*, 2018. 2

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

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

[26] Sara Hahner and Jochen Garcke. Mesh convolutional autoencoder for semi-regular meshes of different sizes. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 885–894, 2022. 1, 2, 3, 6

[27] Sara Hahner, Felix Kerkhoff, and Jochen Garcke. Transfer Learning Using Spectral Convolutional Autoencoders on Semi-Regular Surface Meshes. In *Proceedings of the First*

Learning on Graphs Conference, pages 18:1–18:19. PMLR, 2022. 1, 2, 3, 6, 8

[28] 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

[29] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: A network with an edge. *ACM Transactions on Graphics*, 38:1–12, 2019. 2

[30] Ruqi Huang, Panos Achlioptas, Leonidas Guibas, and Maks Ovsjanikov. Limit shapes - a tool for understanding shape differences and variability in 3D model collections. *Eurographics Symposium on Geometry Processing*, 38:187–202, 2019. 3, 4

[31] S. Joshi, Brad Davis, Matthieu Jomier, and Guido Gerig. Unbiased diffeomorphic atlas construction for computational anatomy. *NeuroImage*, 23:S151–S160, 2004. 3

[32] David G Kendall. A survey of the statistical theory of shape. *Statistical Science*, 4(2):87–99, 1989. 3

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

[34] Clément Lemeunier, Florence Denis, Guillaume Lavoué, and Florent Dupont. Representation learning of 3d meshes using an autoencoder in the spectral domain. *Comput. Graph.*, 107 (C):131–143, 2022. 2

[35] 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, 2022. 2

[36] 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. 2

[37] Or Litany, Alex Bronstein, Michael Bronstein, and Ameesh Makadia. Deformable shape completion with graph convolutional autoencoders. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1886–1895. IEEE, 2018. 1, 2

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

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

[40] 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, 2019. 2, 5

[41] National Crash Analysis Center (NCAC). Finite element model archive (http://web.archive.org/web/20160110143219/www.ncac.gwu.edu/vml/models.html). accesed on: 2016-01-10. 6

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

[43] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps. *ACM Transactions on Graphics*, 31(4):1–11, 2012. 1, 2, 4, 5

[44] 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, 2017. 2

[45] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, pages 604–621. Springer, 2022. 3

[46] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 2

[47] Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2:559–572, 1901. 7

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

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

[50] 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

[51] Can Qin, Haoxuan You, Lichen Wang, C.-C. Jay Kuo, and Yun Fu. Pointdan: A multi-scale 3d domain adaption network for point cloud representation. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 2

[52] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3d faces using convolutional mesh autoencoders. *Proceedings of the European Conference on Computer Vision*, pages 704–720, 2018. 1, 2, 3, 6

[53] 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

[54] Jean-Michel Roufosse, Abhishek Sharma, and Maks Ovsjanikov. Unsupervised deep learning for structured shape matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1617–1627, 2019. 2, 5

[55] 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, 2014. 2

[56] Abhishek Sharma and Maks Ovsjanikov. Weakly supervised deep functional maps for shape matching. In *Advances in Neural Information Processing Systems*, pages 19264–19275. Curran Associates, Inc., 2020. 1, 2, 5

[57] Nicholas Sharp, Souhaib Attaiki, Keenan Crane, and Maks Ovsjanikov. DiffusionNet: Discretization agnostic learning on surfaces. *ACM Transactions on Graphics*, 41(3):1–16, 2022. 1, 5

[58] Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. *ACM Transactions on Graphics*, 23: 399–405, 2004. 6

[59] 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. 2

[60] Qingyang Tan, Lin Gao, Yu Kun Lai, and Shihong Xia. Variational Autoencoders for Deforming 3D Mesh Models. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 5841–5850, 2018. 2

[61] B. Vallet and B. Levy. Spectral Geometry Processing with Manifold Harmonics. *Computer Graphics Forum*, 2008. 4

[62] Fan Wang, Qixing Huang, and Leonidas J. Guibas. Image cosegmentation via consistent functional maps. In *2013 IEEE International Conference on Computer Vision*. IEEE, 2013. 4

[63] Peng-Shuai Wang, Yu-Qi Yang, Qian-Fang Zou, Zhirong Wu, Yang Liu, and Xin Tong. Unsupervised 3d learning for shape analysis via multiresolution instance discrimination, 2021. 3

[64] Jie Yang, Lin Gao, Qingyang Tan, Yi-Hua Huang, Shihong Xia, and Yu-Kun Lai. Multiscale mesh deformation component analysis with attention-based autoencoders. *IEEE Transactions on Visualization and Computer Graphics*, 29: 1301–1317, 2023. 2

[65] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation, 2018. 3

[66] Yu-Jie Yuan, Yu-Kun Lai, Jie Yang, Qi Duan, Hongbo Fu, and Lin Gao. Mesh Variational Autoencoders with Edge Contraction Pooling. In *Conference on Computer Vision and Pattern Recognition Workshops*, pages 1105–1112. IEEE, 2020. 1, 2

[67] Yongheng Zhao, Tolga Birdal, Haowen Deng, and Federico Tombari. 3d point capsule networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

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

[69] Yi Zhou, Chenglei Wu, Zimo Li, Chen Cao, Yuting Ye, Jason Saragih, Hao Li, and Yaser Sheikh. Fully convolutional mesh autoencoder using efficient spatially varying kernels. In *Advances in Neural Information Processing Systems*, pages 9251–9262, 2020. 1, 2, 3, 6