Deep Geometric Functional Maps: Robust Feature Learning for Shape Correspondence: Supplementary Materials

Nicolas Donati
LIX, École Polytechnique
nicolas.donati@polytechnique.edu

Abhishek Sharma
LIX, École Polytechnique
kein.iitian@gmail.com

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

In this document we collect some additional details about the proposed method, and more precisely about the feature extraction layer, an extensive ablation study of the main parts of our method, some complementary quantitative and qualitative results, that due to lack of space were not included in the main manuscript.

1. Additional details on KPConv [8]

Here, we review briefly KPConv method and describe the architecture we used in our implementation.

The input to this network is a 3D point cloud equipped with a signal, such as the 3D coordinates of the points. Let $\mathcal{P} \in \mathbb{R}^{N\times 3}$ be a point cloud in $\mathbb{R}^3$. Let $\mathcal{F} \in \mathbb{R}^{N\times D}$ be a D-dimensional feature signal over $\mathcal{P}$.

The goal of point cloud convolutional networks is to reproduce the architecture of convolutional neural networks on images. It boils down to transferring two key operations on the point cloud structure: the convolution and the pooling operators.

First, we define a convolution between $\mathcal{F}$ and a kernel g at point $x \in \mathbb{R}^3$. Since we only want a signal over the point cloud at each layer, we only need these convolutions at $x \in \mathcal{P}$.

The kernel will be defined as a local function centered on 0 depending on some learnable parameters, taking a D-dimensional feature vector as input and yielding a D'-dimensional feature vector.

More specifically, the kernel is defined in the following way : let r be its radius of action. Let $\mathcal{B}_3^r$ be the corresponding 3d ball. Let K be the number of points, thus the number of parameter matrices in this kernel. Let $\{z_k | k < K\} \subset \mathcal{B}_3^r$ be these points, and $\{W_k | k < K\} \subset \mathcal{M}_{(D,D')}(\mathbb{R})$ be these matrices. Then the kernel g is defined through the formula :

$$g(y) = \sum_{k<K} h(y, z_k) W_k$$

where we simply set $h(y, z) = \max(0, 1 - \frac{\|y-z\|}{\sigma})$, so each point of the kernel has a linear influence of range σ around it.

Method	No Ref	Ref
FMNet	17.	13.
PointNet	18.	14.
Old FMap	4.5	**1.9**
Ours	**3.4**	**1.9**

Table 1. Comparative results for the different ablations of our method.

Then the convolution simply becomes :

$$(\mathcal{F} * g)(x) = \sum_{i | x_i \in \mathcal{B}_3^r} g(x_i - x) f_i$$

Where the learnable parameters are the matrices W_k. We set the points z_k of the kernel to be uniformly organized in $\mathcal{B}_3^r$, so as to better encompass the variations of the convoluted signal at a given point of the point cloud, and a given scale (see [8] supplementaries, Section B for more details).

For the pooling operator, we use a grid sampling that allows us to get the point cloud at an adjustable density. The network can then build hierarchical features over the point clouds by both adjusting the radius of influence of its kernels and the density of the mesh they are performed upon.

Once these two operations are set up, it is easy to build a convolutional feature extractor over the point cloud $\mathcal{P}$. In our work, we use the following architecture :

- Four strided convolutional blocks, each down-sampling the point cloud to half its density, and taking the feature space to another feature space (corresponding to higher-level features) two times larger.
- Four up-sampling layers, taking the signal back on the whole point cloud, through skipping connections followed by 1D convolutions.

2. Ablation study

This section presents the extensive ablation study of all the vital components of our algorithm.

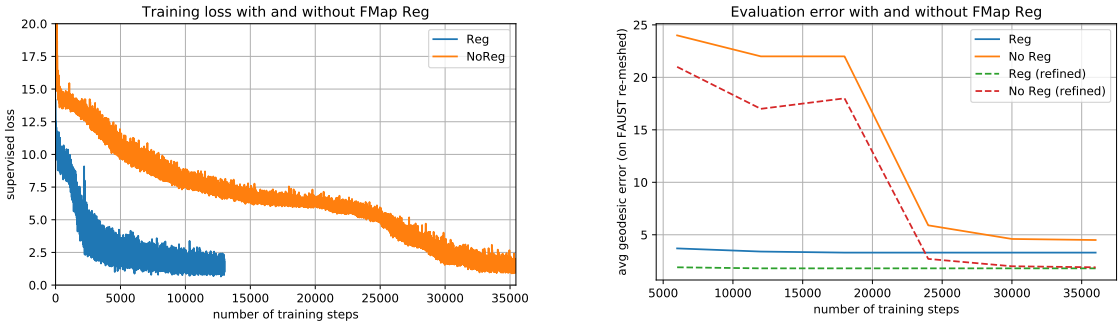

Figure 1. Comparison of convergence speed with and without Laplacian Regularization in the FMap block. *Left*: Training loss evolution, *Right*: Evolution of geodesic error on test set with the number of epochs. Notice how the regularized fmap layer helps drastically with the convergence speed. It gives optimal results within only 500 epochs.

We train all these ablations on 100 random shapes among the 230K proposed by 3D-CODED, as in experiment 2. We test them on the 20 test shapes of FAUST re-meshed, so that the connectivity differs from train to test. The different parts to ablate are :

- The point cloud feature extractor : as explained in the previous sections, it learns descriptors from raw data without relying too much on connectivity. The first ablation consists of our method, but with FMNet [4] feature extractor instead of ours. Similar to FMNet, we use SHOT [9] descriptors. However, we use the same number of eigenvectors as in our method, namely 30. As a general remark, we noticed that lowering this number can often help prevent overfitting in the case of FMNet-based architectures.

- The choice of KPConv [8]. The second ablation study replaces KPConv sampling and feature extractor block with that of PointNet [7]. For this ablation, we use random sampling to 1500 points instead of KPConv grid sampling. Indeed, grid sampling does not provide any guarantee on the number of points after sampling, so it can only be used in a network built to overcome this issue, with batches of adaptable size, which is not the case of PointNet.

- The regularized functional map layer. This third ablation simply consists in replacing our regularized functional map layer by the old functional map layer originally introduced by FMNet. Our layer is in theory mildly heavier than the original one, but in practice for less than 50 eigenvectors the computation times remain the same.

- Lastly, we remove the post-processing refinement step. Here we show the results of every ablation with and without refinement, thus proving it helps in getting better results. As a refinement method, we use the state-of-the-art ZoomOut [5], as mentioned in the main manuscript.

Table 1 shows the ablation study of our method. It demonstrates the importance of all individual blocks and ascertains that all these components are needed to achieve optimal performance with our solution.

However, just looking at the results of the ablation study one does not see the importance of the FMap Reg addition. To prove its efficiency, we compare the learning and evaluation curves of our method, with and without this addition. As can be seen in Figure 1, the models converge much faster with our regularized functional map layer. The models are trained on 100 shapes of the surreal dataset of 3D-CODED as in Experiment 2, and tested on FAUST re-meshed.

In addition, our regularized functional map layer is more robust, and does not result in a Cholesky Decomposition fatal error when computing the spectral map. In comparison, the previous functional map layer gave that fatal error in some experiments, and the model had to be relaunched.

Graphically, as reported in the original functional map paper [6], a natural functional map should be funnel shaped. Our regularized functional map layer naturally computes maps that almost commute with the Laplacians on the shapes. These maps will naturally be close to diagonal matrices (as are funnel shaped maps) in the eigenbasis of the Laplacians, thus reducing the space of matrices attainable by this layer. We believe it helps the feature extractor block focus on setting the *diagonal* coefficients of the functional map as in the ground truth, rather than on trying to get its funnel shape in the first thousand iterations, which is what a model with the original functional map layer does.

3. More quantitative results on Experiments 1 and 2

Figures 2 and 3 summarize the accuracy obtained by our method and some baselines on the different settings

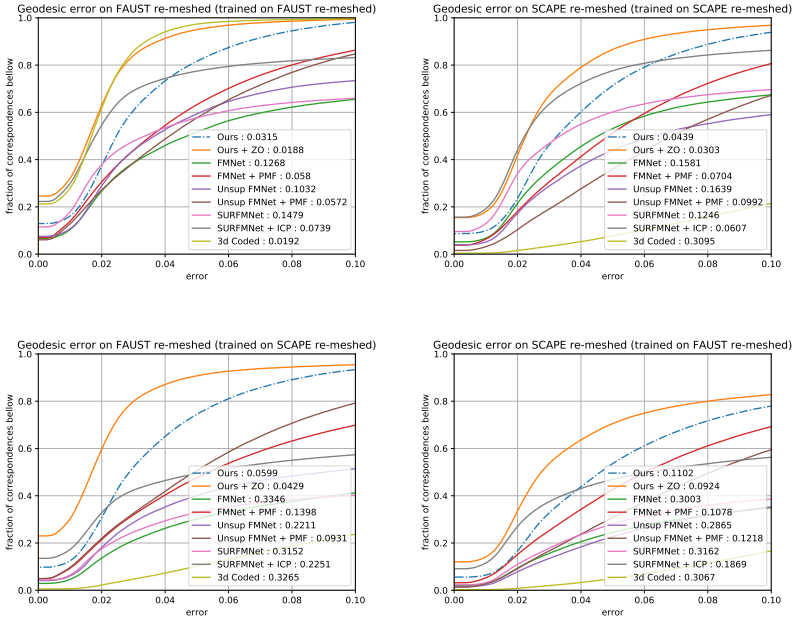

Figure 2. Quantitative results of the different methods using the protocol introduced in [3], on all the settings of Experiment 1

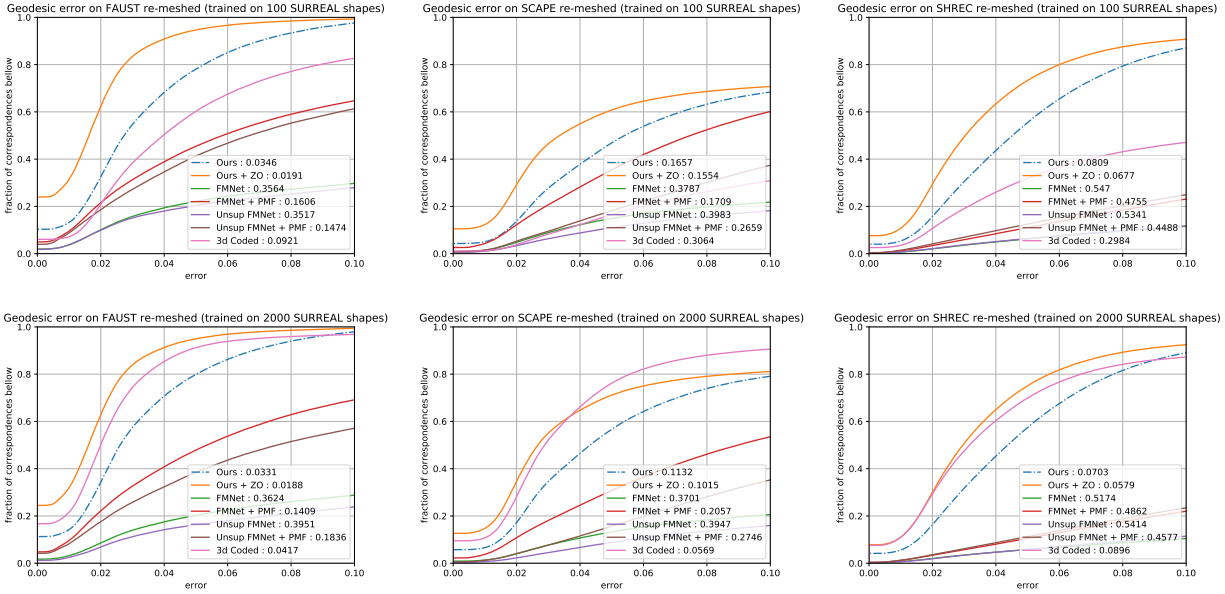

Figure 3. Quantitative results of the different methods using the protocol introduced in [3], on two of the settings of Experiment 2. Top row: 100 shapes (low number). Bottom row: 2000 shapes (high number).

of the two experiments we conducted, using the evaluation protocol introduced in [3]. Note that in all cases but one (trained on 2000 shapes of SURREAL, tested on SCAPE re-meshed), our network achieves the best results even compared to the state-of-the-art methods. As explained more thoroughly in the main manuscript, this proves our method is able to learn point cloud characterizations with only a small amount of data, and by projecting these descriptors in a spectral basis can retrieve accurate correspondences from them. Our method does not need any template and is thus more general than 3D-CODED, in addition to the fact that it trains faster and does not need a big training set.

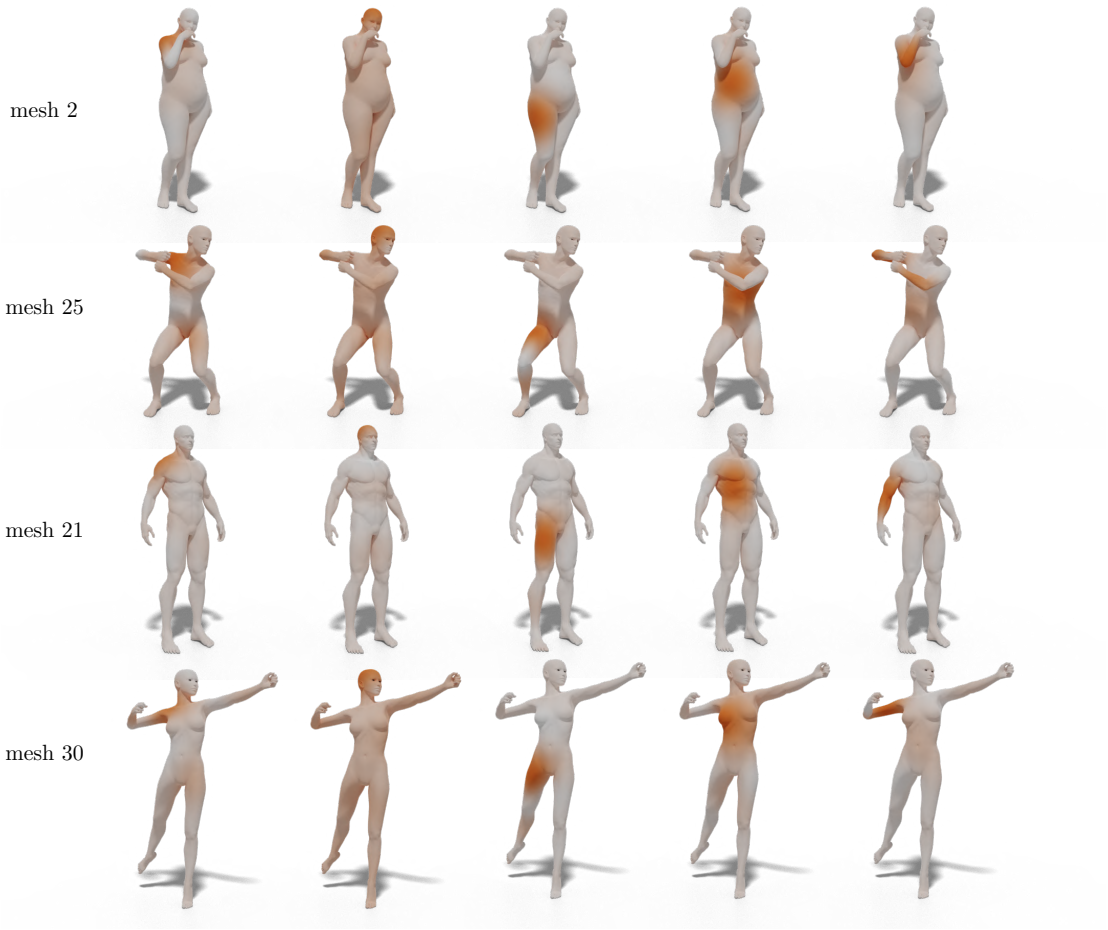

Figure 4. Visualization of spectral descriptors learned by our method (with 2000 surreal shapes) on a test pair of SCAPE re-meshed. The source shape is shown in the first row, and the target shape in the bottom row. Notice how the descriptors are localized and seem to highlight one specific part of the body (first column for shoulder, second for scalp, third for right thigh, fourth for right side of the torso, fifth for elbow).

We believe the superiority of our method with a low number of training shapes shapes is partially due to the fact that 3D-CODED uses a template and operated in the spatial domain, unlike our approach which is template-free, and partly operates in the spectral domain, making it easier to adapt to any new category of 3D shapes.

The relatively low performance of our method on SCAPE in Experiment 2 (see Figure 3) is due to the presence of back-bent shapes in this dataset. These shapes are seen by the network through their truncated spectral approximation, as discussed in section 4, making it unable to exploit refined features such as the face or hands, that could help getting descriptors able to differentiate left from right. Consequently, as there are no back-bent shapes in the training sets of this experiment, these shape are often mapped with a left-to-right symmetry, resulting in a huge error for these particular shapes, increasing the mean error for the whole SCAPE test set.

4. Visualization of some descriptors learned by our method

Our method aims at building descriptors on both input shapes (that are often labeled source and target shapes) from their raw point cloud data. These descriptors are then projected on the eigen basis of the respective Laplace-Beltrami operators of the source and the target shapes. We output these projections, that we call spectral descriptors, and we visualize some of them in Figure 4.

It is remarkable that the descriptors learned on a parametric dataset such as the one used in 3D-CODED still generalize well to shapes with entirely different mesh connectivity and number of points. This is made possible by two components of our method. Firstly, it down-samples the input shapes through grid sampling before building these descriptor functions with convolutional neural networks. This allows for regularity in the input point clouds at all different

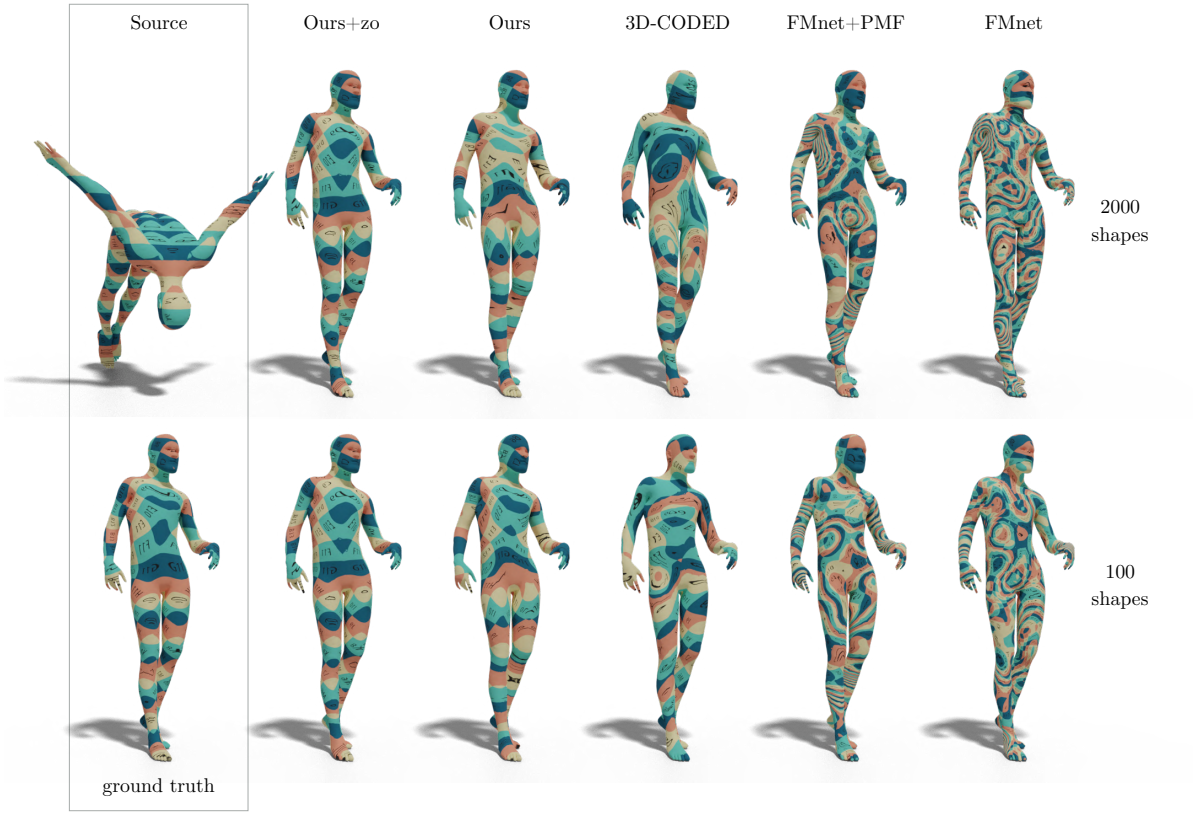

Figure 5. Qualitative results on Experiment 2 through texture transfer, showing cases where our method is the only one that can give good correspondence with only 100 training shapes.

hierarchies (see Figure 6 for an example of such grid sampling). Secondly, the spectral projections take these point cloud descriptions to the shapes intrinsic space, adding some comprehensive surface-related information *without depending too much on the connectivity*, like with SHOT descriptors. Without this intrinsic translation, the network could have trouble differentiating two geometric components close in euclidean space, such as for instance the arms in mesh 25 of Figure 4.

Additionally, these descriptors seem to capture some segmentation information, such as for instance head, arms, body and legs for humans, as can be observed in Figure 4. More precise or complex descriptors such as hand or facial descriptors can not appear with only 30 eigen vectors. It is due to the fact that a spectral reconstruction of a human shape with only 30 eigenvectors does not show small details such as hands, feet or facial features. One would need to push the number of eigenvectors above 100 to see such descriptors appear, and used correctly by the algorithm to produce even better correspondences. However, this could also more easily lead to overfitting.

5. Additional Texture transfer on SHREC'19 re-meshed

In Figure 5 are shown additional qualitative results of our method (with and without Zoomout refinement [5]), 3D coded [1], FMNet [4] and Unsupervised FMNet [2] (with and without PMF refinement [10]), trained on respectively 2000 and 100 shapes, as presented in the Results section, Experiment 2, of the main manuscript.

These results show again the failure of FMNet, due to the change in connectivity. It can be seen more thoroughly in the quantitative graphs provided in Figure 3.

This pair of shapes in Figure 5 represents a challenging case for both 3D-CODED and our method. Indeed, *these networks are not rotation invariant*, as discussed in the implementation section of the main manuscript. Here, the source shape is bent over and its head is really low compared with the rest of the body. 3D-CODED and our method are made robust to rotation around the vertical axis through data augmentation, but here the source shape is slightly rotated along another axis. As we can see, this resulted in poor

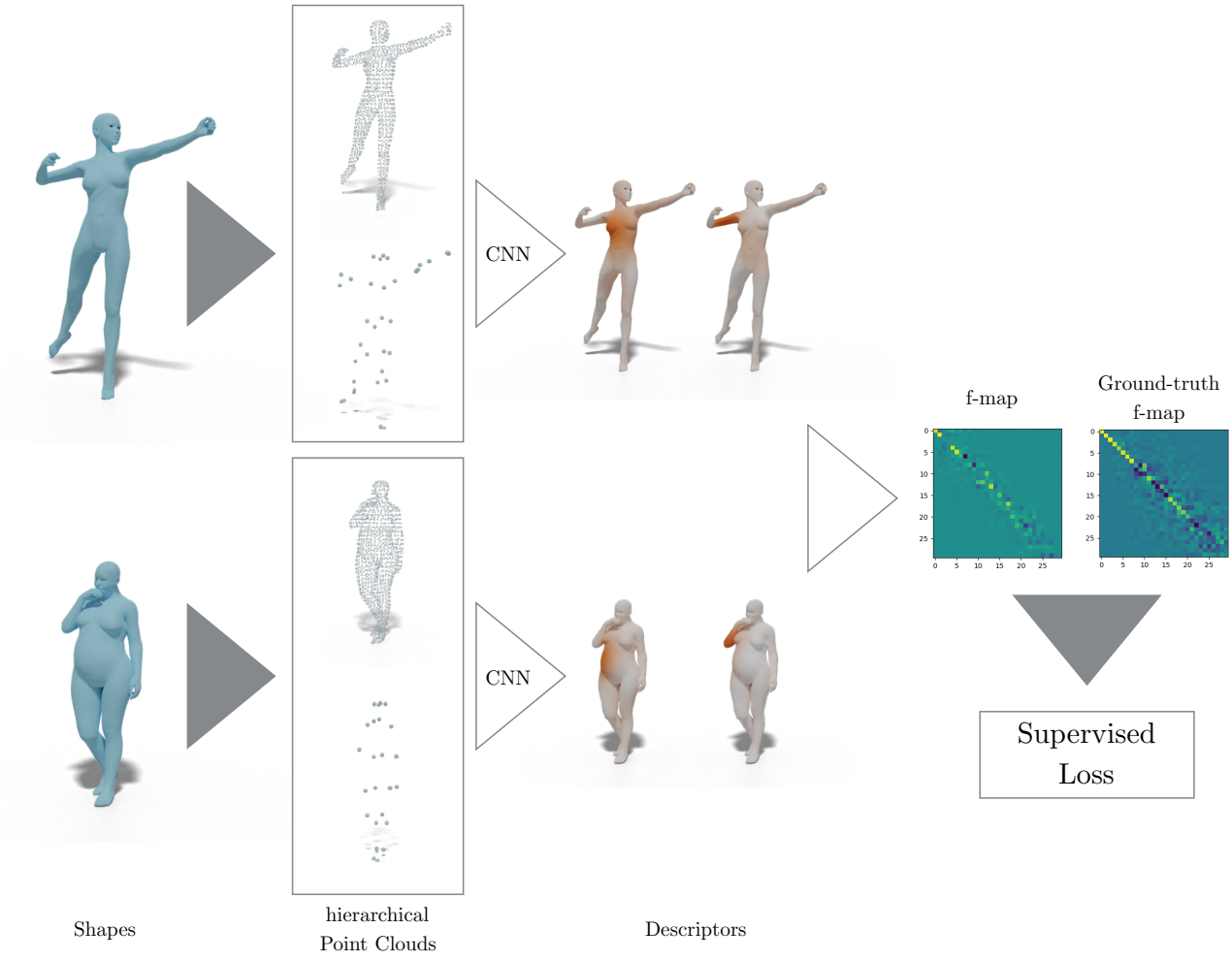

Figure 6. Pipeline of our method: 1) Down-sample source and target shapes with grid sampling (providing the pooling at different scales). 2) Learning point cloud characterizations and project them in the Laplace-Beltrami eigen basis. 3) Compute the functional map from source and target spectral descriptors, with our regularized fmap layer. 4) Compute the loss by comparing the computed functional map with the ground truth map.

reconstructions in the case of 3D-CODED algorithm, even with 2000 training shapes, whereas our method was able to yield good results with both a high and a low number of shapes.

6. General Pipeline

We also provide a visual illustration of our general pipeline in Figure 6 to complement the textual description of our method provided in the main manuscript.

References

[1] 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. 5

[2] Oshri Halimi, Or Litany, Emanuele Rodol'a, Alex Bronstein, and Ron Kimmel. Unsupervised learning of dense shape correspondence. In *CVPR*, 2019. 5

[3] Vladimir G Kim, Yaron Lipman, and Thomas Funkhouser. Blended intrinsic maps. In *ACM Transactions on Graphics (TOG)*, volume 30, page 79. ACM, 2011. 3

[4] Or Litany, Tal Remez, Emanuele Rodolà, Alexander M. Bronstein, and Michael M. Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5660–5668, 2017. 2, 5

[5] Simone Melzi, Jing Ren, Emanuele Rodola, Maks Ovsjanikov, and Peter Wonka. Zoomout: Spectral upsampling for efficient shape correspondence. *ACM Transactions on*

Graphics (Proc. SIGGRAPH Asia), 2019. 2, 5

[6] 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

[7] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. CVPR*, pages 652–660, 2017. 2

[8] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 1, 2

[9] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In *International Conference on Computer Vision (ICCV)*, pages 356–369, 2010. 2

[10] Matthias Vestner, Zorah Lähner, Amit Boyarski, Or Litany, Ron Slossberg, Tal Remez, Emanuele Rodola, Alex Bronstein, Michael Bronstein, Ron Kimmel, and Daniel Cremers. Efficient deformable shape correspondence via kernel matching. In *Proc. 3DV*, 2017. 5