

Supplementary Materials: Instant recovery of shape from spectrum via latent space connections

Riccardo Marin

University of Verona

riccardo.marin.01@univr.it

Emanuele Rodolà

Sapienza University of Rome

rodola@di.uniroma1.it

Arianna Rampini

Sapienza University of Rome

rampini@di.uniroma1.it

Maks Ovsjanikov

Ecole Polytechnique, IP Paris

maks@lix.polytechnique.fr

Umberto Castellani

University of Verona

umberto.castellani@univr.it

Simone Melzi

Ecole Polytechnique, IP Paris

Sapienza University of Rome

melzi@di.uniroma1.it

Abstract

In this document we collect some additional details about the proposed method, architecture and results, that due to lack of space were not included in the main manuscript.

1. Architecture

Meshes. When \mathcal{X} is discretized as a mesh (typical in graphics and geometry processing applications), the AE architecture is a multilayer perceptron (MLP) with a bottleneck and dimensions: $n \times 3 \rightarrow 300 \rightarrow 200 \rightarrow 30 \rightarrow 200 \rightarrow n \times 3$; each layer, except for the last one, is followed by tanh activation. The maps π and ρ are both parametrized by a MLP with dimensions: $k \rightarrow 80 \rightarrow 160 \rightarrow 320 \rightarrow 640 \rightarrow 320 \rightarrow 160 \rightarrow 80 \rightarrow k$; each layer, except for the last one, is followed by batch norm and SeLU activation. We use batch size 16, and the Adam optimizer with learning rate equal to 10^{-4} . Note that we also use this architecture for 2D contours, which are discretized as (2-regular) cycle graphs.

Unorganized point clouds. In this case we do *not* assume the training data to have a consistent vertex labeling. To tackle this setting, we use a PointNet [6] encoder and a fully connected decoder¹; we use the MLP from the mesh case for the decoder. The encoder consists in a shared MLP network on each point with layer output sizes 64 and 128; each layer is followed by batch norm. A maxpool layer is then used to output a 128-dimensional vector, which is reduced with a tanh-activated MLP to dimensions: $128 \rightarrow 64 \rightarrow 30$.

The full implementation of the proposed architecture

¹As suggested by the authors of [6] in <https://github.com/charlesq34/pointnet-autoencoder>

and the data that we used in our experiments can be retrieved here: <https://github.com/riccardomarin/InstantRecoveryFromSpectrum>.

2. Analysis of increasing bandwidth (k)

We performed an experimental analysis for different values of k (the number of Laplacian eigenvalues given as input). This parameter affects both the amount of information used by the network (*i.e.*, more eigenvalues) and the capability of the learned representation (*i.e.*, higher-dimensional latent space). We evaluate our model with $k = 15$, $k = 30$ and $k = 60$ on the shape-from-spectrum reconstruction task, using exactly the same dataset and experimental setup as in Table 1 of the main manuscript.

The quantitative results are reported in Table 1. While cutting k by half produces a sensible performance degradation, we observed that doubling it achieves just slightly better results. For this reason, in our experiments we settled for a value of $k = 30$. From a qualitative perspective, our results suggest that without changing other architecture details, a larger value for k can achieve better precision on the high-frequency details (*e.g.*, the pose of the mouth); as shown in Fig. 1.

	full res	1000	500	200
$k = 30$	1.61	1.62	1.71	2.13
$k = 15$	3.74	3.78	3.72	3.59
$k = 60$	1.60	1.52	1.79	2.05

Table 1. Shape-from-spectrum reconstruction comparisons with different k trained for the same number of epochs; we report average error over 100 shapes of an unseen subject from the COMA dataset [7]. All errors must be rescaled by 10^{-5} .

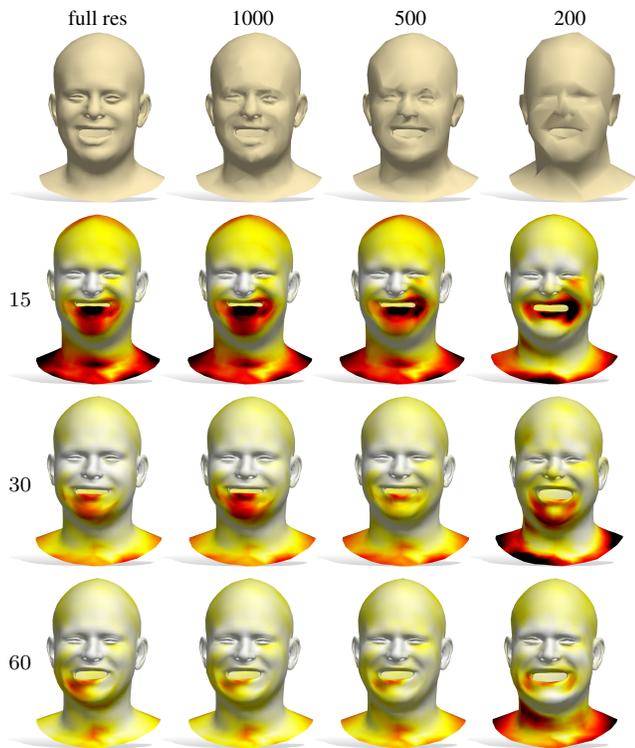


Figure 1. Examples of shape-from-spectrum reconstructions at different values for the spectral bandwidth k ; see Table 1 for a quantitative evaluation.

3. Laplace-Beltrami operator discretization

One of the advantages of our pipeline is that we can use the *cubic* FEM (e.g. [8, Sec. 4.1]) to discretize the Laplace-Beltrami operator, virtually without any additional cost if compared with the linear finite elements. The cubic FEM yields a more accurate discretization and improves the quality of the results produced by our pipeline. In Fig. 2 we collect additional comparisons between the two different discretizations.

4. Training and test details

Here we summarize the data involved in the trainings and in the experiments. In the code that we will release upon acceptance, we will attach pre-processing scripts and when possible the data used to permit replication.

Overall, we perform five different trainings: for CoMa and SMAL datasets we train the dense and the PointNet version of our Network on the same set of data. We obtained the point clouds for PointNet trainings with a sparse sampling over the original meshes (the number of sampled point is around 20% of the mesh vertices at full resolution): some examples of training point clouds are shown in Figure 3. For the ShapeNet dataset, only the PointNet version is trained using 500 points for shape. The applications use

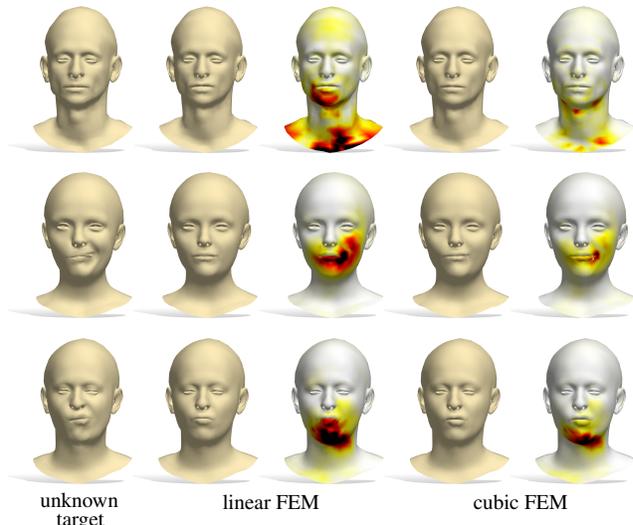


Figure 2. Additional comparisons (one per row) between the use of eigenvalues of the Laplacian discretized with linear FEM or with cubic FEM in our shape-from-spectrum pipeline. The heatmap encodes point-wise reconstruction error, growing from white to dark red. “Unknown target” in the left-most column refers to the source shape from which the eigenvalues are computed.

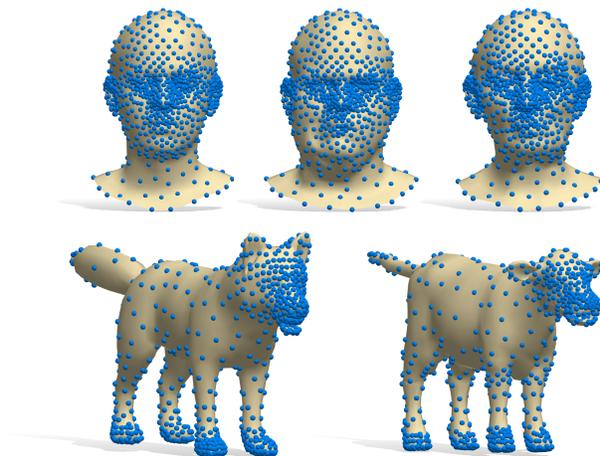


Figure 3. Examples of point clouds used to train the PointNet version of our network in case of CoMa (first row) and SMAL (second row).

these five models, and we do not do any ad-hoc optimization for different tasks.

Concerning the involved training sets:

CoMa. We use 1,853 shapes randomly picked between 11 subjects from the CoMa dataset [7] (168 shapes for each subject), leaving out the 12th subject for testing purpose.

SMAL. We use 4,430 animal meshes generated with the SMAL [9] generative model, among all the five possible classes. Thanks to the method explained in the original paper, we can cluster them in the correct races while we ex-

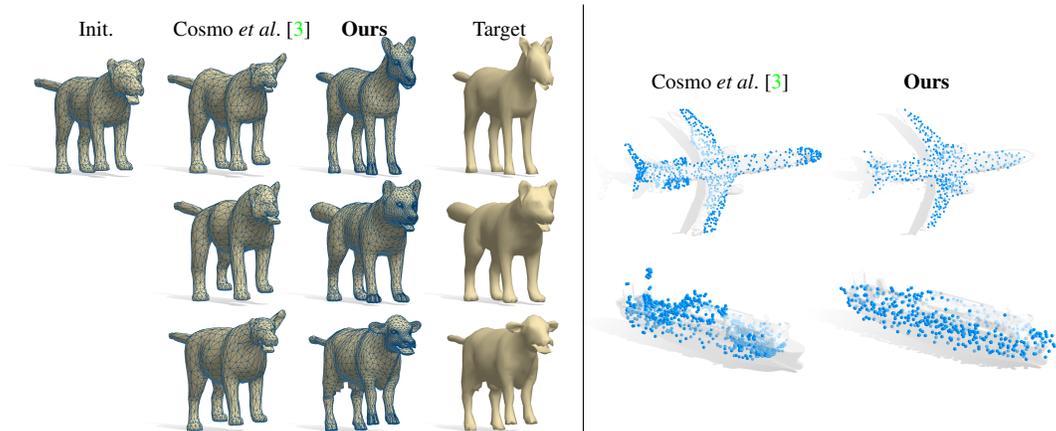


Figure 4. Comparison in shape from spectrum estimation between isospectralization [3] and our method in the case of meshes (left) and point clouds (right). The *Init.* shape on the left is the neutral shape of the SMAL[9] generative model used as initialization of isospectralization algorithm for animals. The target shapes in the point cloud case are shown under the point clouds.

plore the inter-class variety.

ShapeNet. We would remark that ShapeNetCore dataset [2] is not practical for standard spectral analysis since the models consist of non-manifold disconnected pieces. For this reason, we used a processed watertight version of it [4]. From it, we pruned the meshes that still have artifacts (e.g. disconnected components). Finally, we remeshed the remaining ones to have at maximum 10K vertices. For training, we rely on four different classes: airplanes, boats, screens, and chairs. We pick 2,047 meshes from these classes, among the ones with a more reliable spectrum (i.e. removing outliers case). This last property was not considered for the test set, to challenge ourselves also against non-standard models (as you can see in Figure 11 of the main manuscript, where the target airplane has a tail’s part missing).

Concerning the quantitative experiments:

Shape from spectrum. This experiment was performed on a total of 400 shapes; 100 shapes of the unseen subject from the CoMa dataset at four different resolutions.

Estimating point cloud spectra. Here we consider a total of 1,186 models for the test: 393 shapes of different subjects from FLAME [5] dataset at two different samplings, and 400 shapes from ShapeNet from the four classes.

Shape matching. Finally, for the shape matching, we consider 100 couples of random animals of different species. These animals are generated with the same method of the above training set, but with higher variance in the generative space of SMAL. In this way, the testset presents more extreme characteristics of the ones seen at training time. For matching, we mainly consider extreme cases where high non-isometric deformation occurs (e.g. as shown in Figure 11 between hippos and horses). As a final comment, we would emphasize that our results against Nearest-Neighbor

suggest our generalization capability and the real challenge of our testsets.

5. Shape from spectrum

In the task of recovering shapes from Laplacian spectra we compared with the *isospectralization* approach introduced in [3]. In Fig. 4, we report some qualitative examples from SMAL dataset [9] and ShapeNet dataset [4]. For SMAL, both the unknown shape to be recovered and our reconstruction are meshes (left panel of Fig. 4); for objects from ShapeNet, our network has been trained on unorganized sparse point clouds, so the reconstruction in this case is a point cloud as well (right panel of Fig. 4). Our method recovers shape from spectrum in a *single forward pass*, while isospectralization *deforms* an initial shape such that its spectrum aligns the one of a target shape using geometric regularizers. As initialization for isospectralization we used the template of SMAL [9] for animals, and the closest (in terms of distance between the eigenvalues) shape from our training set for objects of ShapeNet. Even so, isospectralization produces unrealistic instances or fails to capture the geometry of the target compared to our method.

6. Super-resolution

Our trained network model is largely insensitive to mesh resolution and sampling as we show in our experiments in the main manuscript. This property makes our network appropriate for the task of mesh super-resolution. Given a low-resolution mesh as input, we can recover a higher resolution counterpart of the mesh that is also in dense point-to-point correspondence with models from the training set. We can do that in a single shot starting only from the spectrum of the input low-resolution mesh. In Fig. 5, we show a com-

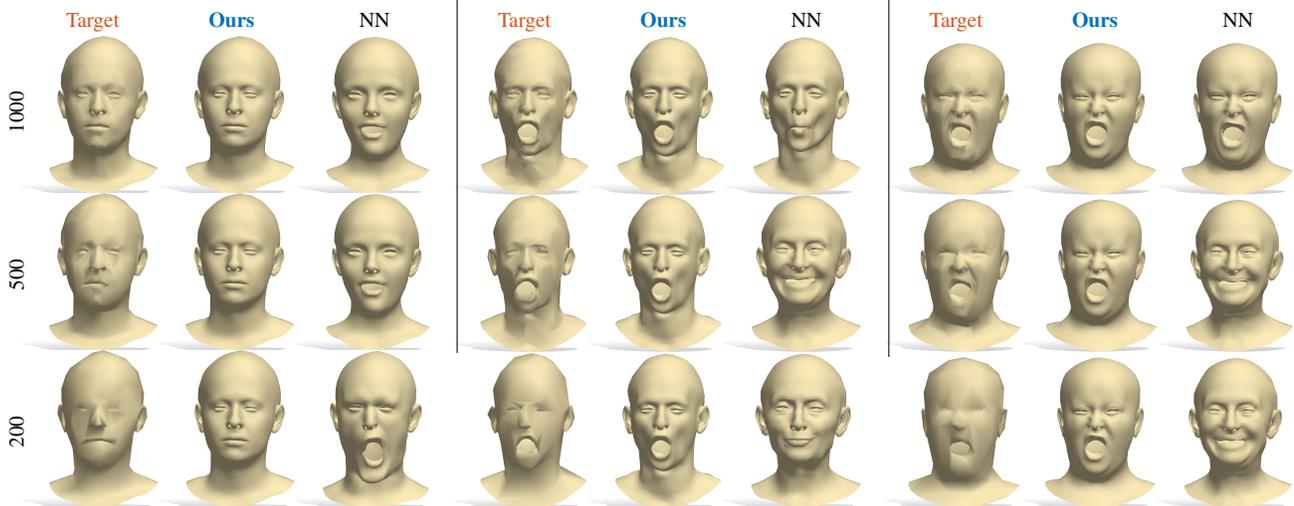


Figure 5. Mesh super-resolution for input shapes at decreasing resolution (top to bottom, target shapes have respectively 1000, 500 and 200 vertices). Our solutions match closely with the original high-resolution version of the input shapes, while the nearest neighbor baseline (NN) predicts the wrong identity or pose.

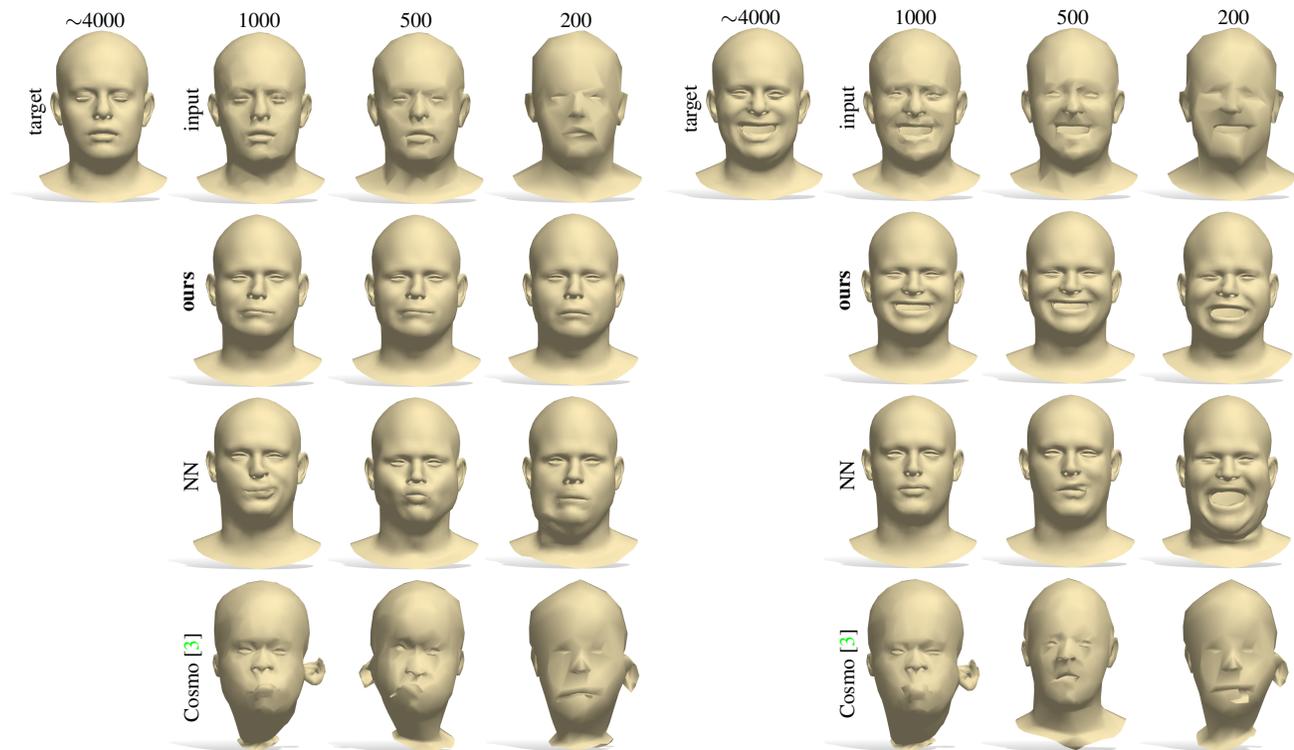


Figure 6. Two mesh super-resolution examples for input shapes at decreasing resolution (top row of each of the two images, left to right). Our solutions match closely with the original high-resolution version of the input shapes (top left), while other approaches either predict the wrong pose (NN baseline) or generate an unrealistic shape (Cosmo *et al.*).

parison with nearest-neighbors (NN) between eigenvalues (with shapes within the training set), and the *isospectralization* method of Cosmo *et al.* [3] for two different shapes from a subject that is not involved in the training of our net-

work model.

In Fig. 6, we perform the same experiment on three different shapes never seen during the training, and belonging to a subject whose other poses are in the training set. This

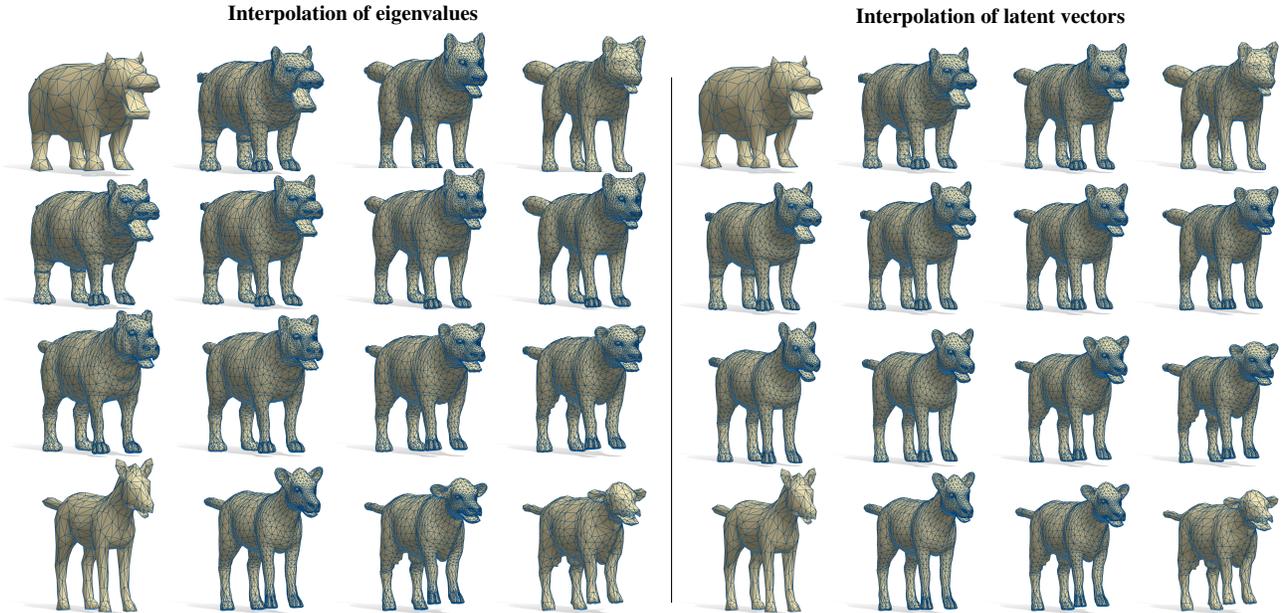


Figure 7. Comparison between interpolation in the space of eigenvalues (left) and the interpolation in the latent space (right). The input shapes are four low-resolution versions of as many shapes from the training set.

comparison highlights that our method generalizes better than NN also for shapes that should be well described by the training set. This allows us to state that navigation driven by the eigenvalues is more accurate than NN navigation of the latent space.

7. Spectral shape exploration

In the main manuscript, we show two different sub-tasks of shape exploration: shape interpolation and interactive spectrum-driven exploration.

As we claim in the main manuscript, an alternative for shape interpolation is to regard the eigenvalues themselves as a space that we can explore, as is typically done with latent spaces. In Fig. 7, on the same four low-resolution shapes, we compare interpolation in the latent space (on the right) with interpolation in the space of the eigenvalues (on the left). The former evaluation is also included in the main manuscript. For the latter, given the four shapes, we first compute their spectra, directly perform a bilinear interpolation among these spectra, and finally reconstruct the corresponding shapes. Also in this case, the entire procedure can be performed for shapes with different connectivity. The linear interpolation among spectra gives rise to meaningful results. However, observe that the space of spectra (where each point is a different sequence of eigenvalues) is not a proper vector space, and taking linear steps is not completely appropriate in this domain.

We support this with two additional figures. In Fig. 8, for four shapes in the training set, we provide a denser linear interpolation in the space of the eigenvalues. In Fig. 9, we

show the same linear interpolation between the spectra of a pair of shapes that come from two different datasets, and which were never seen in the training phase. Both cases confirm that interpolation can indeed be performed in the space of eigenvalues.

Finally, in a video attached to these supplementary materials, we show a demo of interactive spectrum-driven shape exploration. In the video, we modify the eigenvalues of a given shape and we show how these modifications allow us to interactively navigate the space of shapes.

8. Style transfer

As described in the main manuscript, we apply our trained network for the style transfer task. In Fig. 10 we show some additional examples. We emphasize here that our method is correspondence-free, since the input eigenvalues completely encode the target style. Our method also does not rely on the presence of the undeformed source shape, used in previous analogies applications, e.g. [1] that phrased the problem of finding X such that $(A \rightarrow B, C \rightarrow X)$ for some known A, B, C . These examples confirm the robustness and accuracy of the proposed method in this application.

References

- [1] D. Boscaini, D. Eynard, D. Kourounis, and M. M. Bronstein. Shape-from-operator: recovering shapes from intrinsic operators. *Computer Graphics Forum*, 34(2):265–274, 2015.
- [2] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su,

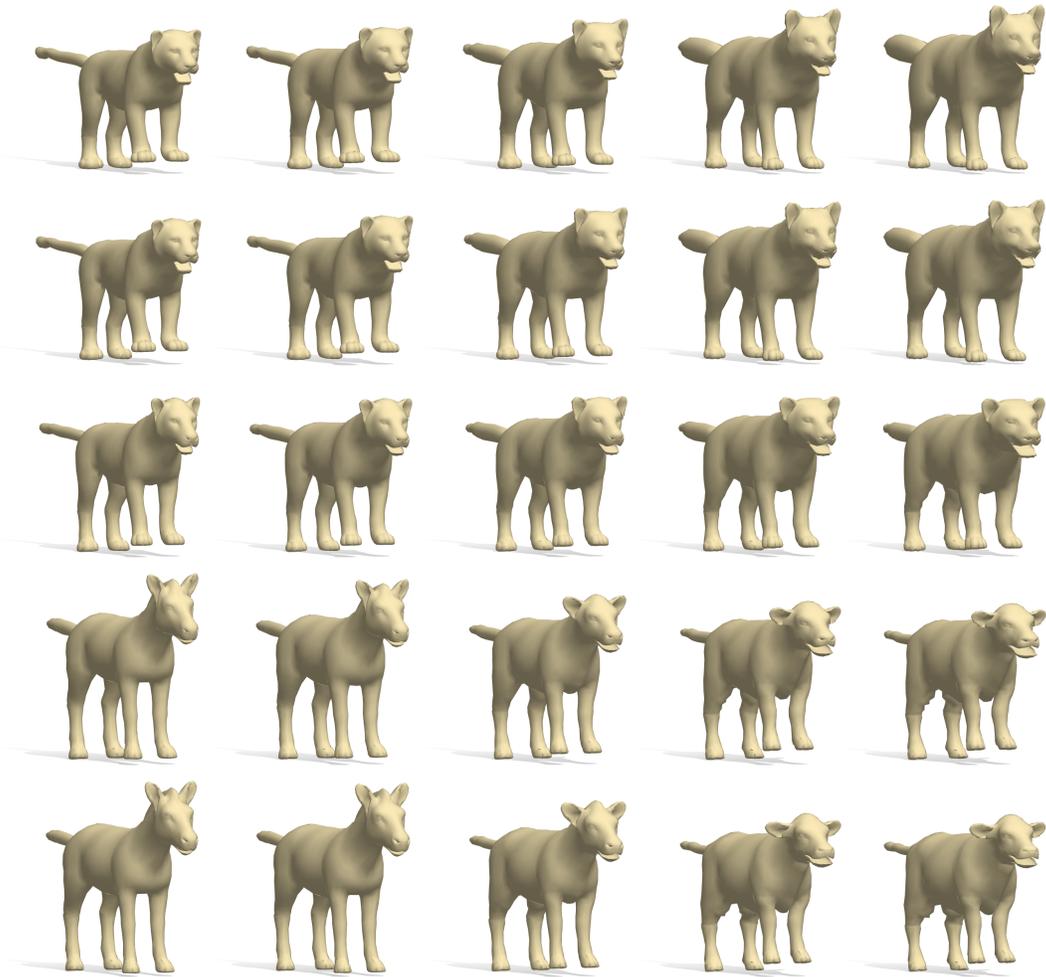


Figure 8. Interpolation *between spectra* (as opposed to latent vectors) between four shapes in the training set.

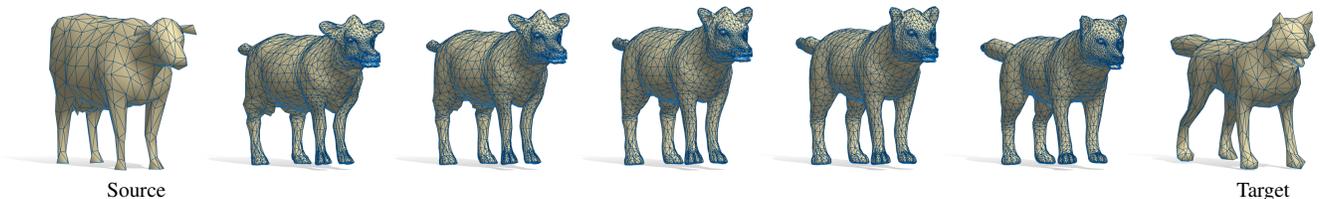


Figure 9. Interpolation between the spectra of two low-poly shapes that belong to two different datasets, never seen at training time.

- J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.
- [3] L. Cosmo, M. Panine, A. Rampini, M. Ovsjanikov, M. M. Bronstein, and E. Rodolà. Isospectralization, or how to hear shape, style, and correspondence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7529–7538, 2019.
- [4] J. Huang, H. Su, and L. Guibas. Robust watertight manifold surface generation method for shapenet models. *arXiv preprint arXiv:1802.01698*, 2018.
- [5] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), 2017.
- [6] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.
- [7] A. Ranjan, T. Bolkart, S. Sanyal, and M. J. Black. Generating 3D faces using convolutional mesh autoencoders. In *European*

Conference on Computer Vision (ECCV), 2018.

- [8] M. Reuter. Hierarchical shape segmentation and registration via topological features of laplace-beltrami eigenfunctions. *International Journal of Computer Vision*, 89(2-3):287–308, 2010.
- [9] S. Zuffi, A. Kanazawa, D. Jacobs, and M. J. Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

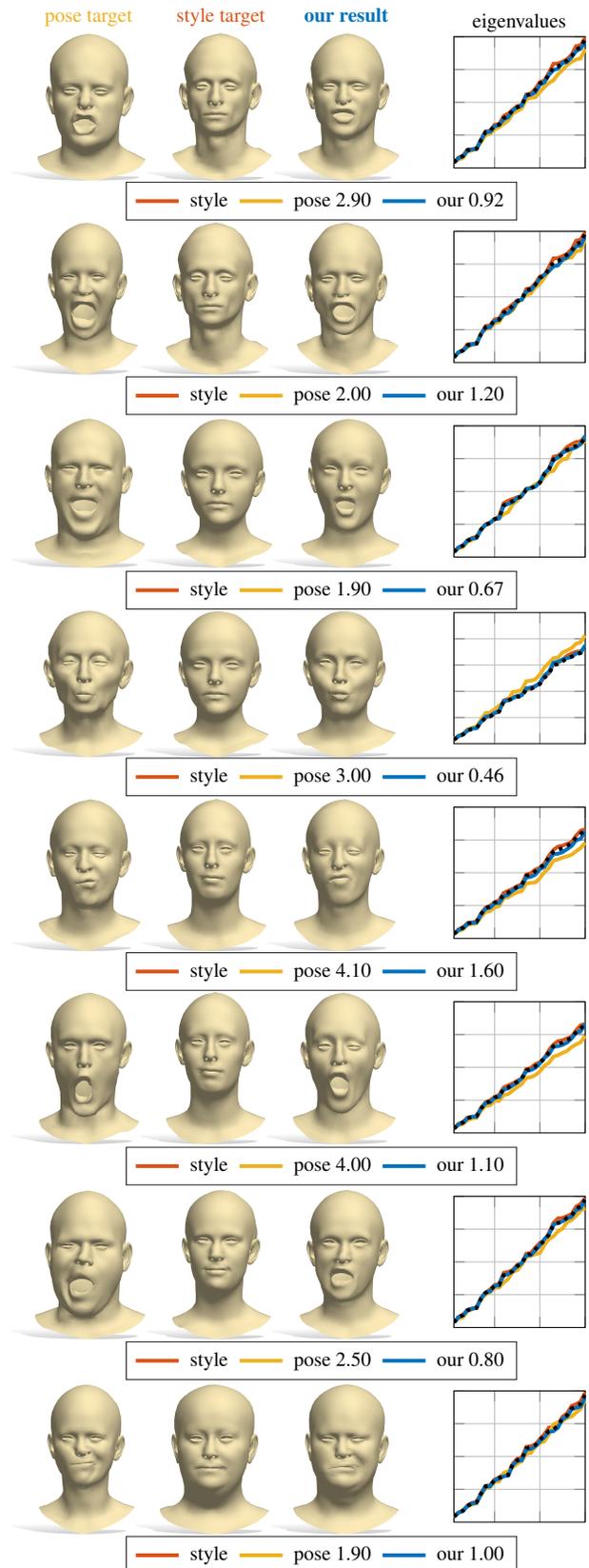


Figure 10. Examples of style transfer. The target style (middle) is applied to the target pose (left), obtaining our result (right).