Intrinsic Point Cloud Interpolation via Dual Latent Space Navigation

Marie-Julie Rakotosaona
LIX, Ecole Polytechnique
mrakotos@lix.polytechnique.fr

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

Abstract

We present a learning-based method for interpolating and manipulating 3D shapes represented as point clouds, that is explicitly designed to preserve intrinsic shape properties. Our approach is based on constructing a dual encoding space that enables shape synthesis and, at the same time, provides links to the metric on the shape, which is typically not available on point cloud data. Our method works in a single pass and avoids expensive optimization, employed by existing techniques. Furthermore, the strong regularization provided by our dual latent space approach also helps to improve shape recovery in challenging settings from noisy point clouds across datasets. Extensive experiments show that our method results in more realistic and smoother interpolations compared to baselines.

1. Introduction

A core problem in 3D computer vision is to analyze, encode and manipulate shapes represented as point clouds. Point clouds are particularly useful compared to other representations due to their generality, simplicity and independence from complex data-structures that might be necessary to handle e.g. triangle meshes or dense voxel grids. For all of these reasons, and with the introduction of PointNet [32] and its variants, point clouds have also gained popularity in machine learning applications, including point-based generative models.

Unfortunately the flexibility of point cloud representations also comes at a cost, as they do not encode any topological or metric information of the underlying surface. Thus, methods trained on point cloud data can by their nature be insensitive to metric distortion that might appear on generated shapes. This problem is particularly prominent in 3D shape interpolation, where a common approach is to generate intermediate shapes by interpolating the learned latent vectors. In this case, even if the end-shapes are realistic, the intermediate ones can have severe distortions that are very difficult to detect and correct using only point-based information. More generally, several works have observed that latent spaces of generative models built on point cloud data can fail to capture the space of natural shapes. This makes it difficult to control or navigate them while maintaining realism of generated results.

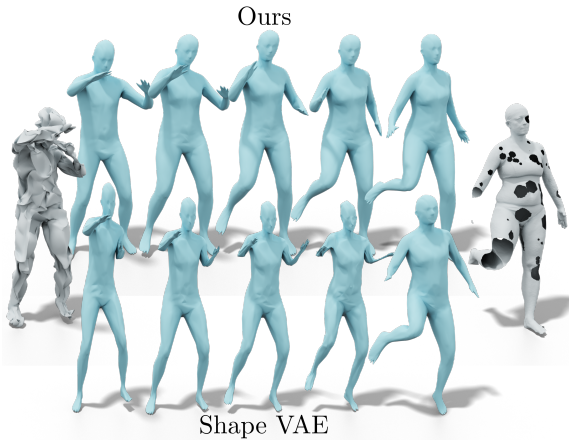

Figure 1. Intrinsic point cloud interpolation between points from a noisy mesh (left, reconstructed in first blue column) and points from an incomplete scan with holes (right, reconstructed in last blue column). Our method both reconstructs the shape better and produces a more natural interpolation than a PointNet-based Shape Variational Auto-Encoder (Shape VAE).

In this paper, we introduce a novel architecture aimed specifically at injecting intrinsic information into a generative point-based network. Our method works by learning cycle-consistent mappings across the latent space obtained by a point cloud auto-encoder and another feature encoding that captures the intrinsic shape metric. We show that these two parts can be optimized jointly using shapes represented as triangle meshes during training. The resulting linked latent space combines the strengths of a generative latent model, and the intrinsic metric structure. Finally, we use the learned networks at test time on raw 3D point clouds that are neither in correspondence with the training shapes,

nor contain any connectivity information.

Our approach is general and not only leads to smooth interpolations, while avoiding expensive iterative optimization, but also, as we show below, leads to more accurate shape reconstruction from noisy point clouds across different datasets. We demonstrate on a wide range of experiments that our approach can significantly improve upon recent baselines in terms of the accuracy and smoothness of the interpolation and enables a range of novel applications.

2. Related Work

Shape interpolation, also known as morphing in certain contexts, and exploration is a vast and well-researched area of computer vision and computer graphics (see [27] for a survey of the early approaches) and its full overview is beyond the scope of this paper. Below we review works most closely related to ours, and concentrate, in particular, on either structure-preserving mesh interpolation techniques, or recent learning-based methods that focus on point clouds.

Classical methods for 3D shape interpolation have primarily focused on designing well-founded geometric metrics, and associated optimization methods that enable smooth structure-preserving interpolations. Early works in this direction include variants of as-rigid-as-possible interpolation and modeling [2, 23, 42] and various *representations* of shape deformation that facilitate specific transformation types, e.g. [39, 21, 29, 12, 38] among many others.

A somewhat more principled approach is provided by the notion of *shape spaces* [24, 31] in which interpolation can be phrased as computing a shortest path (geodesic). In the case of surface meshes, this approach was studied in detail in [25] and then extended in numerous follow-up works, including [41, 13, 20, 19] among many others. These approaches enjoy a rich theoretical foundation, but are typically restricted to shapes having a fixed connectivity and can lead to difficult optimization problems at test time.

We also note a recent set of methods based on the formalism of *optimal transport* [6, 36, 10] which have also been used for shape interpolation. These approaches treat the input shapes as probability measures that are interpolated via efficient optimization techniques.

Somewhat more closely related to ours are data-driven and feature-based interpolation methods. These include interpolation based on hand-crafted features [15, 22] or by analyzing various *local* shape spaces obtained by analyzing a shape collection [16, 43, 40]. These approaches work well if the input shapes are sufficiently similar, but require triangle meshes and dense point-wise correspondences, or a single template that is fitted to all input data to build a statistical model, e.g. [18, 7, 8].

Most closely related to ours are recent generative models that operate directly on point clouds [1, 28, 30]. Approaches in this area are largely inspired by the seminal work of PointNet and its variants [32, 33] and are most commonly based on learned *latent* feature encoder and a decoder networks that learn to recover the 3D shape from its latent vector. Despite significant progress in this area, however, the structure of learned latent spaces is typically not easy to control or analyze. For example, it is well-known (see e.g. [22]) that commonly used linear interpolation in latent space can give rise to unrealistic shapes that are difficult to detect and rectify.

Common approaches to address these issues include extensive data augmentation [17], using adversarial losses that aim to penalize unrealistic instances [28, 5] or modifying the metric in the latent space, e.g. by computing the Jacobian of the mapping to the embedding space [11, 35] or using feature-based metrics at test time [26, 14]. Unfortunately, as we show below such techniques either lead to difficult optimization problems at test time, or can still result in significant shape distortion.

Contribution In this paper, we propose to address this challenge by building a *dual latent space* that combines a learned feature encoding used for a point-based generative model, similar to [1] with another parallel encoding that aims to capture the intrinsic shape metric given by the lengths of edges of triangle meshes only required during training. This second encoding exploits the intuition provided by mesh-based interpolation techniques [25, 19, 34] that highlight the importance of *interpolating the intrinsic metric* rather than the point coordinates. We achieve this by constructing dense networks that allow to "translate" between the two latent spaces, and enable smooth and accurate interpolation at test time without requiring input point clouds to be in correspondence or solving expensive optimization problems.

3. Method

3.1. Overview

Our main goal is construct a generative model for point cloud data that is also informed by the intrinsic structure of the underlying surfaces. To this end we combine a VAE architecture that learns a low-dimensional latent space for the given shape class with an auto-encoder that learns latent codes capturing the lengths of edges of the underlying triangle mesh. Importantly, the triangle mesh structure is only required during training. At test time, we use the point cloud VAE and the associated trained mapping networks that allow us to navigate across the two latent spaces. This allows us to compute both a generative latent code and the corresponding metric-based latent encoding, associated with a given point cloud.

Figure 2 gives an overview of our network. It consists of three main building blocks and training steps: a point cloud

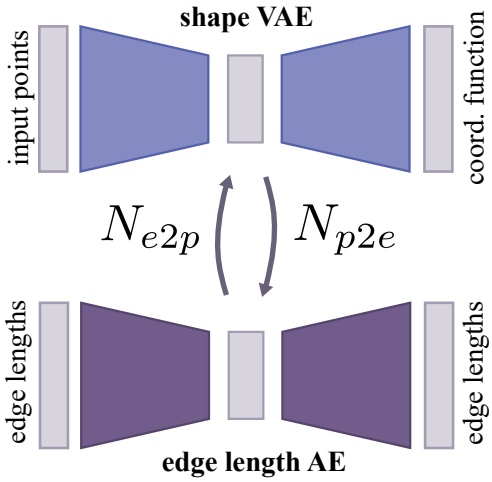

Figure 2. Architecture of our model. We build a mapping between a shape generative model (shape VAE) and an edge length auto-encoder therefore constructing a dual latent space that enables shape synthesis (top) while preserving intrinsic shape information (bottom).

VAE, an auto-encoder of the intrinsic shape metric, and two "translation" or mapping networks that enable communication between the two latent spaces. These networks are used at test time to endow given point clouds with intrinsic metric structure which is then used, in particular, for more accurate point cloud interpolation. We assume that the training data is given in the form of triangle meshes with fixed connectivity, while the input at test time consists of unorganized point clouds. In the following section we describe our architecture and the associated losses, while the implementation and experimental details are given in Section 4.

3.2. Architecture

Shape auto-encoder. Our first building block (Figure 2 top) consists of a PointNet-based VAE. We denote the encoder and decoder functions as enc_p and dec_p respectively. We describe the implementation in further details in Section 3.4.

We promote rotation-invariance, which plays a role when we combine this block with the metric-based latent space, we train this VAE use an invariant reconstruction loss L_{rec} from [22]. Namely, we first compute the optimal rigid transformation R^* between the ground truth P and the predicted point cloud $\tilde{P}$ using Kabsh algorithm [4]. We then compute the mean square error between the coordinates after alignment.

$$L_{rec}(P, \tilde{P}) = \frac{1}{n} \sum_{i=1}^{n} \|R^*(P_i) - \tilde{P}_i\|^2. \tag{1}$$

Here the summation is over the points in the point cloud. As shown in [22] this loss is differentiable using the derivative of the Singular Value Decomposition.

Note that our encoder enc_p inherits the permutation invariance of the PointNet [32], which is crucial in real applications. Namely, this allows us to encode arbitrary point clouds at test time even if they have significantly different sampling and are not in correspondence with the training data.

Metric-based autoencoder As observed in previous works and as we confirm below, the point cloud VAE can capture the structure of individual shapes, but often fails to reflect the overall structure of *shape space*, which is particularly evident in shape interpolation applications. We propose to address this issue by constructing a separate auto-encoder that aims to capture the intrinsic shape metric, and, at the same time to learn mappings across the two latent spaces.

For this, we first build an autoencoder (enc_e, dec_e) with dense layers that aims to reconstruct a list of edge lengths. Note that since we assume 1-1 correspondence at training time, the list of lengths of edges can be given in canonical (e.g., lexicographic with respect to vertex ids) order. We therefore build an auto-encoder that takes encodes this list into a compact vector and decodes it back from the latent representation. Our training loss for this part consists of two components: a mean squared error on the predicted edge lengths and an additional term that promotes linearity in the learned latent space:

$$L_e(E_A) = \|\text{dec}_e(\text{enc}_e(E_A)) - E_A\| \tag{2}$$

$$L_{lin}(E_A, E_B) = \left\| \frac{\text{dec}_e(\text{enc}(E_A)) + \text{dec}_e(\text{enc}(E_B))}{2} - \text{dec}_e\left(\frac{\text{enc}(E_A) + \text{enc}(E_B)}{2}\right) \right\|^2. \tag{3}$$

Here E_A, E_B are the lists of edge lengths corresponding to the triangle meshes given at training A, B given during training. Our general motivation for the second loss L_{lin} is to explicitly encourage linear structure in the latent space. We base our intuition on the observations in previous mesh interpolation and manipulation works [25, 34], which argue for smoothness of the interpolated metric. The linear structure in the edge-based latent space promotes this smoothness, while our mappings to the latent space of the shape auto-encoder maintains the realism of the resulting shapes.

Mapping networks Given the pretrained shape VAE and the metric-based autoencoder, we train two dense mapping networks that translate elements between the two latent spaces. We use N_{p2e} and N_{e2p} to denote the networks that translate an element from the shape (resp. edge) latent space to the edge (resp. metric) latent space.

We use cycle consistent losses to train these networks. Namely:

$$L_{map1}(P) = \alpha \|R^*(\text{dec}_p(N_{e2p}(N_{p2e}(\text{enc}_p(P))))) - P\|^2 + \beta \|el(\text{dec}_p(N_{e2p}(N_{p2e}(\text{enc}_p(P))))) - el(P)\|^2 \quad (4)$$

$$L_{map2}(E) = \|\text{dec}_e(N_{p2e}(N_{e2p}(\text{enc}_e(E)))) - E\|^2, \quad (5)$$

where P and E are a point cloud and a list of edge-lengths given at training respectively, $el()$ a function that computes the list of edge lengths given a mesh at training and R^* is the optimal rotation computed in the same way as in Eq.(1). The edge auto-encoder mapping quality can be measured by a simple reconstruction loss. On the shape VAE, we can either measure a reconstruction loss of point coordinates or an edge lengths reconstruction loss from the recovered points. We can use different combination of these two losses.

Our overall loss is then simply the sum of two terms $L_{map1} + L_{map2}$ for shapes given at training.

Network Training To summarize we train our overall network architecture described in Figure 2 in three separate steps. First we train the shape-based Variational Auto Encoder using the loss given in Eq. (1). Then we train the edge-based auto-encoder using the sum of the losses in Eq. (2) and Eq. (3). Finally we train the dense networks N_{e2p} and N_{p2e} using the sum of the two losses in Eq. (4) and Eq. (5). We also experimented with training the different components jointly but have observed that the problem is both more difficult and the relative properties of the computed latent spaces become less pronounced when trained together, leading to less realistic reconstructions.

3.3. Navigating the restricted latent space

After training the networks as described above, we use them at test time for shape reconstruction and interpolation. We stress that at test time we do not use the edge encoder and decoder networks enc_e, dec_e, as they require canonical edge ordering. Instead we use the shape based auto-encoder and the mapping networks N_{p2e}, N_{e2p} to better preserve intrinsic shape properties.

Our main observation is that the latent space associated with the shape auto-encoder provides a way to synthesize point clouds, while the latent space of the metric-based auto-encoder helps to restrict that space and reduce metric distortion. Therefore, at test time we use a combination of these two latent spaces and the networks N_{p2e}, N_{e2p} to navigate between them.

Interpolation Given two possibly noisy unorganized point clouds P_A and P_B we first compute their associated metric-based latent codes: $m_A = N_{p2e}(\text{enc}_p(P_A))$ and $m_B = N_{p2e}(\text{enc}_p(P_B))$. Here we use the permutation-invariance of our encoder enc_p which allows to relate unorganize point sets. We then perform linear interpolation between m_A and m_B but use the *shape decoder* dec_P for reconstruction. In particular we compute a family of intermediate point clouds as follows:

$$P_\alpha = \text{dec}_p\left(N_{e2p}\left((1-\alpha)m_A + \alpha m_B\right)\right), \ \alpha \in [0..1] \quad (6)$$

In other words, we interpolate the latent codes in the metric-based latent space, but perform the reconstruction, via the shape decoder dec_p. This allows us to make sure that the reconstructed shapes are both realistic and their intrinsic metric is interpolated smoothly. Note that unlike the purely geometric methods, such as [25], our approach does not rely on the given mesh structure at test time. Instead, we employ the learned metric-based latent space as a proxy for recovering the intrinsic shape structure, which as we show below, is sufficient to obtain accurate and smooth interpolations.

The edge auto-encoder being fully rotation invariant, it is necessary to align the output shapes at test time. We can do so easily by using the same optimal rigid transformation R^* Eq. (1).

3.4. Implementation Details

Datasets We train our networks on two different datasets: humans and animals. For humans, we use the dataset proposed in [22]. The dataset contains 17440 shapes subsampled to 1k points from DFAUST [9] and SURREAL [37]. The testset contains 10 sub-collections (character + action sequence, each consisting of 80 shapes) that are isolated from the training set of DFAUST and 2000 shapes from SURREAL dataset. During training the area of each shape is normalized to a common value. For animals we sample 12000 shapes from the SMAL dataset [44]. We sample an equal number of shapes from the 5 categories (lions, tigers, horses, cows, hippos, dogs) to build a training set of 10000 shapes and a testset of 2000 shapes. We simplify the shapes from SMAL to 2002 points per mesh.

Shape VAE The Shape VAE uses a PointNet based autoencoder to which we concatenate 2 MLP layers in order to generate a distribution to draw the latent representation from. We set the dimension of the latent space to 256. The decoder is a simple fully connected layer of depth 3. We train the model using the loss from Eq. (1) and the standard KL divergence term.

Edge auto-encoder The edge auto-encoder is a fully connected network with 7 layers. The encoder contains 4 layer. It encodes the input edge lengths to a latent representation

	EL (10^{-5})	PC (10^{-4})	area (10^{-8})
Shape VAE	3.158	**3.693**	2.592
Edge AE	3.127	-	-
Ours L_{map1e}	**1.282**	14.896	**1.150**
Ours L_{map1p}	2.476	**3.451**	2.062
Ours $L_{map1e+p}$	**1.606**	3.837	**1.372**

Table 1. Mean squared reconstruction losses on the humans test-set. Edge length reconstruction loss (EL), Point cloud coordinates reconstruction loss (PC) and per triangle area difference

of size 256. The decoder uses 3 MLP layers to reconstruct the original edge lengths.

Mapping networks We build each mapping network N_{p2e} and N_{e2p} with 3 MLP layers of size 256.

Finally we implemented the presented architecture using Tensorflow and used the Adam optimizer for training. Our complete implementation will be released soon.

4. Results

4.1. Shape reconstruction

In our first experiment, we evaluate the reconstruction accuracy of our model on the DFAUST/SURREAL test-set. In this case, we simply use the combination of the trained shape VAE with the networks N_{p2e} and N_{e2p} for shape reconstruction from unorganized point clouds. In other words, to reconstruct a given point cloud we use the composition $\text{dec}_p \circ N_{e2p} \circ N_{p2e} \circ \text{enc}_p$ and compare it to various alternatives.

Table 5 shows the reconstruction accuracy of the different networks. We measure the point coordinate reconstruction and the edge length accuracy with respect to the ground truth mesh. We train different versions of our architecture with different weights on the two components of the loss L_{map1} described in Section 3.2. We denote by L_{map1e} the part that enforces good edge reconstruction and by L_{map1p} the one for good point coordinates. We observe that for any combination of these losses our architecture improves the reconstruction quality of the intrinsic information of the underlying mesh such as edge lengths and per triangle area compared to the basic Shape VAE (which uses the path $\text{dec}_p \circ \text{enc}_p$) and edge auto-encoder. In the rest of our experiments, we will mostly use the models using L_{map1e} and $L_{map1e+p}$ for their generalizing and geometric qualities.

We further evaluate the generalization capacity of our network by evaluating on the SCAPE [3] dataset. For testing we sample 1000 random points from the surface of each mesh. Table 2 shows an improvement in the reconstruction for our method. We observe even higher relative performance when comparing the total volume and total area of the reconstructed shapes which give a sense of the perceived quality of the shapes. Shape distortions are often related to

	CD (10^{-3})	MS volume (10^{-5})	MS area
Shape VAE	4.703	30.851	0.1382
Ours $L_{map1p+e}$	4.135	9.47	0.047

Table 2. Reconstruction accuracy on the SCAPE dataset. We measure the Chamfer distance (CD), mean square total volume difference and MS total area difference

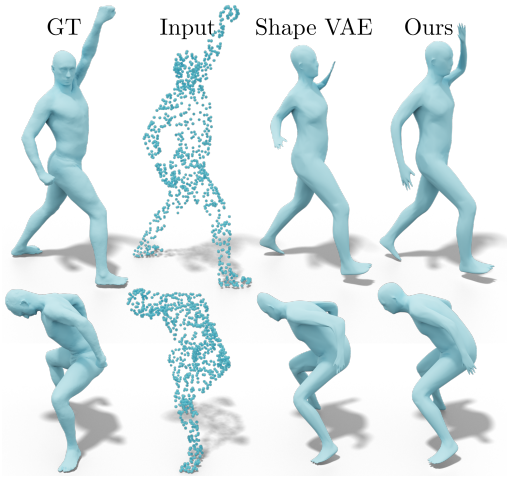

Figure 3. Shape reconstruction from SCAPE. We reconstruct from 1k random points on the surface.

shrunk or disproportional body parts. Figure 3 shows two examples of reconstruction from the dataset. While the simple Shape VAE, is still able to reconstruct the overall position of the tested human, the output has distortions near the hands (top) and the legs (bottom). Our method generated more natural meshes even though the dataset is completely unknown with an entirely different underlying mesh, different body type and poses that are not the same as those seen at training.

Finally, as shown in Figure 1, our method is robust to high levels of noise (left), holes, and missing parts (right).

4.2. Shape interpolation

Next, we evaluate our method on our core application of shape interpolation and compare it against six different recent baselines. Namely, we compare to simple linear interpolation in the latent space of our Shape VAE, an auto-encoder proposed in 3D-CODED [17] and a PointNet++ [33] based autoencoder that we train on the same dataset with the same latent space dimension. We explore some of the ideas from [35] [11] by building a method that minimizes the linear interpolation in the latent space path length using L2 loss in shape space (GD L2). We also compare to method that optimizes the edge length difference loss on the interpolated path (GD EL). Finally we compare to a non data-driven method simplified from [25] (GD Coord.). In this method we first build an initial path by linearly interpo-

	EL	area (10^{-4})	volume (10^{-4})
PC VAE	0.3760	1.830	1.402
3D-CODED [17]	0.6130	3.137	1.243
GD L2 [35] [11]	0.3631	1.838	1.483
GD EL	0.2985	1.714	1.703
GD Coord. [25]	**0.0345**	**0.248**	**0.152**
Pointnet++ [33]	0.2993	1.586	335.2
Ours L_{map1e}	**0.2234**	1.293	**0.318**
Ours $L_{map1e+p}$	0.2301	**1.220**	0.385

Table 3. We report the mean squared variance of the edge length (EL), per surface area and total shape volume over the interpolations of 100 shape pairs. We highlight that among data-driven methods that result in realistic shape reconstruction, our method achieves lowest variance across all intrinsic features.

lating the source and target shape points coordinates. Then we optimize for the variance of edge lengths directly by performing gradient descent on the sum of squared differences of edge lengths along the interpolation path with respect to the points positions.

Remark that GD Coord., GD L2 and GD EL methods all rely on gradient descent to compute each interpolation *at test time*. In other words, these approaches all require to solve a highly non-trivial optimization problem during interpolation. On top of the extra time and computational cost of that optimization, there are additional parameters (learning rate, number of iterations) that have to be hand picked by the user for each case to get the best results. By interpolating directly in the learned edge length space and using the computed mappings between the two spaces, our method avoids the extra computation while still respecting the intrinsic shape structure.

To evaluate the interpolations we sample 50 shapes from the DFAUST testset using farthest point sampling. We then test on 100 random pairs from those 50 shapes.

Table 3 shows quantitative comparisons. Given an interpolation path (P_n) obtained by each method, we compute the mean squared variance of various features f on the path. We consider three shape features: lengths of edges, overall surface area and overall volume enclosed by the shape (computed from the mesh embedding). For each of these, we compute the sum of the squared differences across all instances in the interpolating sequence:

$$V_f(P_n) = \frac{1}{n-1} \sum_{i=1}^{n-1} \|f(P_{i+1}) - f(P_i)\|^2. \qquad (7)$$

Intuitively, we expect a good interpolation method to result in smooth interpolations which would have low variance across all of the intrinsic shape properties.

To be fair when comparing with PointNet++ since it was trained on normalized bounding boxes and not area, we normalize the total area of each output. The large volume variance of this baseline is primarily due to bad reconstruction quality of the source and target shapes.

	edge length	area (10^{-3})	volume (10^{-2})
PC VAE	2.068	3.742	2.754
Ours $L_{map1e+p}$	1.538	2.975	1.728

Table 4. MS variance on 100 pairs from 50 shapes obtained by farthest points sampling on animals dataset (SMAL)

As shown in Table 3 our method has overall the best performance among these intrinsic methods except from GD Coord. where the edge length is directly optimized with gradient descent. At the same time, as this method is purely driven by optimizing edge length variance it is unaware of the structure of shape space and as shown in Figure 5 results in highly unrealistic interpolating instances. This also suggests that the highly complex multi-resolution (both in time and in space) optimization scheme used in [25] is essential for obtaining high quality results with that approach, as it is a purely geometric and not a data-driven method. Instead, our method obtains high quality interpolations in one shot and is capable of handling unorganized point clouds as input.

In Figure 4 we compare linear interpolations in the original shape VAE latent space and linear interpolations in the edge auto-encoder latent space. Our method generates more natural looking bodies (first two rows) and even interpolates well between shapes pairs where the end results differs highly from the linear interpolation of the coordinates (last two rows).

In Figure 5 we illustrate the interpolated shapes between the input source and target, shown in grey. As highlighted above, we notice that while GD Coord. has low variance in the interpolated intrinsic features, the reconstructed shapes do not look natural. Overall, our method presents less distortions and more smooth interpolations compared to all baselines. We present more comparisons and evaluations in the supplementary materials.

We further evaluate our model on the SMAL dataset. To build the interpolation pairs from the testset, we sample 10 shapes per category by farthest points sampling. We then choose 100 random pairs from that dataset. In Figure 6 we show the results of interpolating between two horses. We observe that linear interpolation in the shape VAE latent space leads to shape distortions such as shorter legs (middle) and wrong shape size estimation (top left). Table 4 shows a comparison of interpolations between the shape VAE and our approach. Similarly to above, our method shows improvement at interpolating intrinsic information.

5. Conclusion, Limitations & Future Work

We presented a method for interpolating unorganized point clouds. Key to our approach is a dual latent space en-

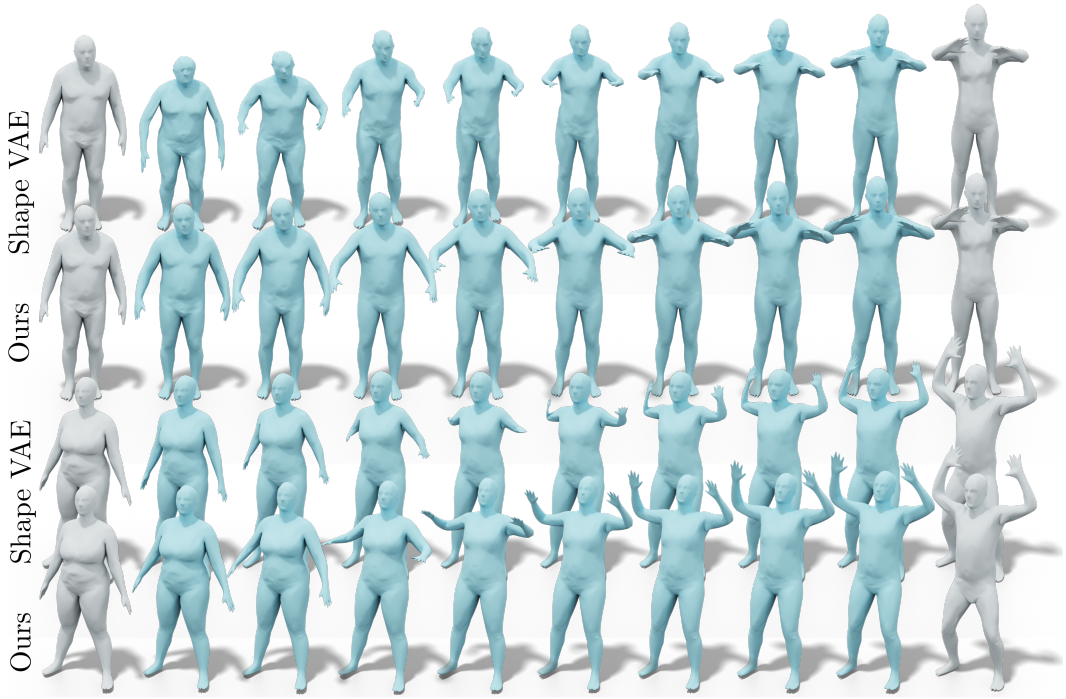

Figure 4. We compare linear interpolations in the original shape VAE latent space and linear interpolation in edge auto-encoder latent space.

coding that both captures a the overall shape structure and the intrinsic metric, given by edge lengths provided during training. We demonstrate that our approach leads to significant improvement compared to existing methods, both in terms of interpolation smoothness and quality of the generated results.

In the future, we plan to extend our method to also incorporate other features such as semantic classes or segmentations. Our method is also currently restricted to categories for which training shapes share the same connectivity. Finally, it would also be interesting to explore the utility of our dual encoding space in other applications, on images and other data representations.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 40–49, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. 2

[2] Marc Alexa, Daniel Cohen-Or, and David Levin. As-rigid-as-possible shape interpolation. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 157–164. ACM Press/Addison-Wesley Publishing Co., 2000. 2

[3] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: shape completion and animation of people. In *ACM transactions on graphics (TOG)*, volume 24, pages 408–416. ACM, 2005. 5

[4] K Somani Arun, Thomas S Huang, and Steven D Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on pattern analysis and machine intelligence*, 1(5):698–700, 1987. 3

[5] Heli Ben-Hamu, Haggai Maron, Itay Kezurer, Gal Avineri, and Yaron Lipman. Multi-chart generative surface modeling. In *SIGGRAPH Asia 2018 Technical Papers*, page 215. ACM, 2018. 2

[6] Jean-David Benamou and Yann Brenier. A computational fluid mechanics solution to the monge-kantorovich mass transfer problem. *Numerische Mathematik*, 84(3):375–393, 2000. 2

[7] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. Faust: Dataset and evaluation for 3d mesh registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3794–3801, 2014. 2

[8] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J Black. Dynamic faust: Registering human bodies in motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6233–6242, 2017. 2

[9] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: Registering human bod-

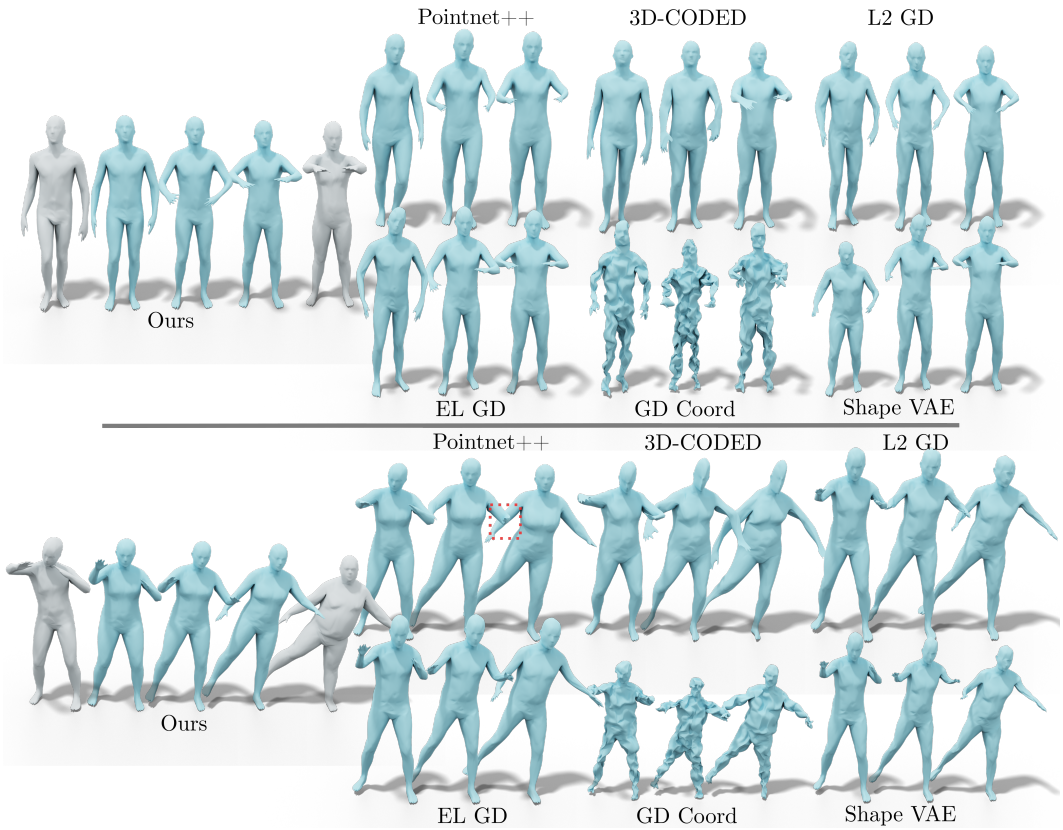

Figure 5. Qualitative comparison of interpolation on DFAUST testset

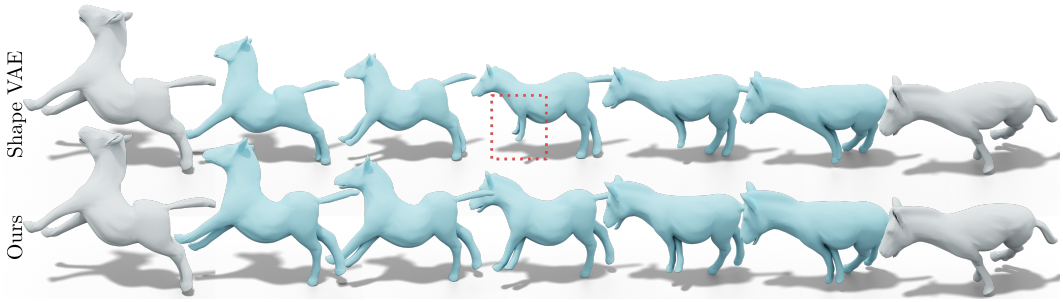

Figure 6. Interpolation of two horses from SMAL dataset

ies in motion. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 4

[10] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51(1):22–45, 2015. 2

[11] Nutan Chen, Alexej Klushyn, Richard Kurle, Xueyan Jiang, Justin Bayer, and Patrick van der Smagt. Metrics for deep generative models. *arXiv preprint arXiv:1711.01204*, 2017. 2, 5, 6

[12] Keenan Crane, Ulrich Pinkall, and Peter Schröder. Spin transformations of discrete surfaces. *ACM Transactions on Graphics (TOG)*, 30(4):104, 2011. 2

[13] Oren Freifeld and Michael J Black. Lie bodies: A manifold representation of 3d human shape. In *European Conference on Computer Vision*, pages 1–14. Springer, 2012. 2

[14] Max F Frenzel, Bogdan Teleaga, and Asahi Ushio. Latent space cartography: Generalised metric-inspired measures and measure-based transformations for generative models. *arXiv preprint arXiv:1902.02113*, 2019. 2

[15] Lin Gao, Shu-Yu Chen, Yu-Kun Lai, and Shihong Xia. Data-driven shape interpolation and morphing editing. *Computer Graphics Forum*, 36(8):19–31, 2017. 2

[16] Lin Gao, Yu-Kun Lai, Qi-Xing Huang, and Shi-Min Hu. A

data-driven approach to realistic shape morphing. *Computer graphics forum*, 32(2pt4):449–457, 2013. 2

[17] 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. 2, 5, 6

[18] Nils Hasler, Carsten Stoll, Martin Sunkel, Bodo Rosenhahn, and H-P Seidel. A statistical model of human pose and body shape. *Computer graphics forum*, 28(2):337–346, 2009. 2

[19] Behrend Heeren, Martin Rumpf, Peter Schröder, Max Wardetzky, and Benedikt Wirth. Splines in the space of shells. *Computer Graphics Forum*, 35(5):111–120, 2016. 2

[20] Behrend Heeren, Martin Rumpf, Max Wardetzky, and Benedikt Wirth. Time-discrete geodesics in the space of shells. *Computer Graphics Forum*, 31(5):1755–1764, 2012. 2

[21] Jin Huang, Xiaohan Shi, Xinguo Liu, Kun Zhou, Li-Yi Wei, Shang-Hua Teng, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Subspace gradient domain mesh deformation. *ACM Transactions on Graphics (TOG)*, 25(3):1126–1134, 2006. 2

[22] Ruqi Huang, Marie-Julie Rakotosaona, Panos Achlioptas, Leonidas Guibas, and Maks Ovsjanikov. Operatornet: Recovering 3d shapes from difference operators. *arXiv preprint arXiv:1904.10754*, 2019. 2, 3, 4

[23] Takeo Igarashi, Tomer Moscovich, and John F Hughes. As-rigid-as-possible shape manipulation. *ACM transactions on Graphics (TOG)*, 24(3):1134–1141, 2005. 2

[24] David G Kendall. Shape manifolds, procrustean metrics, and complex projective spaces. *Bulletin of the London Mathematical Society*, 16(2):81–121, 1984. 2

[25] Martin Kilian, Niloy J Mitra, and Helmut Pottmann. Geometric modeling in shape space. *ACM Transactions on Graphics (TOG)*, 26(3):64, 2007. 2, 3, 4, 5, 6

[26] Samuli Laine. Feature-based metrics for exploring the latent space of generative models. 2018. 2

[27] Francis Lazarus and Anne Verroust. Three-dimensional metamorphosis: a survey. *The Visual Computer*, 14(8):373–389, 1998. 2

[28] Chun-Liang Li, Manzil Zaheer, Yang Zhang, Barnabas Poczos, and Ruslan Salakhutdinov. Point cloud GAN. *arXiv preprint arXiv:1810.05795*, 2018. 2

[29] Yaron Lipman, Daniel Cohen-Or, Ran Gal, and David Levin. Volume and shape preservation via moving frame manipulation. *ACM Transactions on Graphics (TOG)*, 26(1):5, 2007. 2

[30] Xinhai Liu, Zhizhong Han, Xin Wen, Yu-Shen Liu, and Matthias Zwicker. L2g auto-encoder: Understanding point clouds by local-to-global reconstruction with hierarchical self-attention. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 989–997. ACM, 2019. 2

[31] Peter W Michor and David Bryant Mumford. Riemannian geometries on spaces of plane curves. *Journal of the European Mathematical Society*, 2006. 2

[32] 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. 1, 2, 3

[33] 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 5099–5108, 2017. 2, 5, 6

[34] Josua Sassen, Behrend Heeren, Klaus Hildebrandt, and Martin Rumpf. Solving Variational Problems Using Nonlinear Rotation-invariant Coordinates. In David Bommes and Hui Huang, editors, *Symposium on Geometry Processing 2019-Posters*. The Eurographics Association, 2019. 2, 3

[35] Hang Shao, Abhishek Kumar, and P Thomas Fletcher. The riemannian geometry of deep generative models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 315–323, 2018. 2, 5, 6

[36] Justin Solomon, Fernando De Goes, Gabriel Peyré, Marco Cuturi, Adrian Butscher, Andy Nguyen, Tao Du, and Leonidas Guibas. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Transactions on Graphics (TOG)*, 34(4):66, 2015. 2

[37] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *CVPR*, 2017. 4

[38] Amir Vaxman, Christian Müller, and Ofir Weber. Conformal mesh deformations with möbius transformations. *ACM Transactions on Graphics (TOG)*, 34(4):55, 2015. 2

[39] Wolfram Von Funck, Holger Theisel, and Hans-Peter Seidel. Vector field based shape deformations. *ACM Transactions on Graphics (TOG)*, 25(3):1118–1125, 2006. 2

[40] Philipp von Radziewsky, Elmar Eisemann, Hans-Peter Seidel, and Klaus Hildebrandt. Optimized subspaces for deformation-based modeling and shape interpolation. *Computers & Graphics*, 58:128–138, 2016. 2

[41] Benedikt Wirth, Leah Bar, Martin Rumpf, and Guillermo Sapiro. A continuum mechanical approach to geodesics in shape space. *International Journal of Computer Vision*, 93(3):293–318, 2011. 2

[42] Dong Xu, Hongxin Zhang, Qing Wang, and Hujun Bao. Poisson shape interpolation. *Graphical models*, 68(3):268–281, 2006. 2

[43] Zhibang Zhang, Guiqing Li, Huina Lu, Yaobin Ouyang, Mengxiao Yin, and Chuhua Xian. Fast as-isometric-as-possible shape interpolation. *Computers & Graphics*, 46:244–256, 2015. 2

[44] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael J. Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 4

Supplementary

A. Shape reconstruction

As mentioned in the main manuscript, our approach not only enables better interpolation, but also results in more accurate reconstructions from noisy input. Here we provide

	EL (10^{-5})	PC (10^{-4})	area (10^{-8})
Shape VAE	2.740	**2.321**	2.405
Ours $L_{map1e+p}$	**1.666**	2.611	**1.554**
3D-CODED	6.323	5.803	5.485
3D-CODED*	6.284	4.260	5.409
PointNet++	2.835	3.224	2.835

Table 5. Mean squared reconstruction losses on the DFAUST testset. Edge length reconstruction loss (EL), Point cloud rotation invariant reconstruction loss (PC) and per triangle area difference

additional quantitative evaluation of the reconstruction performance of different baseline methods. To be fair to 3D-CODED, we normalize the total area of the output shapes. We evaluate this method before (3D-CODED) and after (3D-CODED*) their additional step of Chamfer Distance minimization. Note that in the case of 3D-CODED* additional optimization *at test time* is required to recompute the latent code that best approximates the input. Our method, like a standard auto-encoder, performs the reconstruction in one shot.

In all of the experiments the training data is the combination of DFAUST and SURREAL datasets, and the test data is the DFAUST test shapes, both with and without noise.

Table 5 shows reconstruction results for several baselines on the DFAUST test shapes. We report the edge length accuracy (EL), rotation-invariant point cloud reconstruction accuracy (PC) and per triangle area reconstruction accuracy (area). Note that our approach achieves the best overall reconstruction accuracy, especially on the intrinsic quantities and gives slightly worse reconstruction extrinsic reconstruction loss (PC) compared to our Shape VAE. Both of these give significantly better accuracy than all other baselines. We provide qualitative examples in Figure 7. Note that our method leads to both preservation of the overall shape structure and significantly less intrinsic distortion compared to all baselines.

Table 6 shows reconstruction performance on noisy point clouds. Each noisy point cloud is built by adding gaussian noise of magnitude 5% of the scale of the mesh to each vertex coordinate. We observe that our method outperforms the other baselines for all the features. Figure 8 shows reconstructed meshes from the noisy point clouds. Notice that our method performs better at recovering the original pose and body type than the different baselines.

Table 7 shows reconstruction results on simplified point clouds. We randomly sample 500 points from the test shapes surfaces. We recall that the network was trained on 1000 point clouds. We observe that our method is more robust to under-sampling. In particular, and contrary to other methods, the intrinsic properties remain competitive with the performance from Table 5.

	EL (10^{-5})	PC (10^{-4})	area (10^{-8})
Shape VAE	6.253	8.978	6.008
Ours $L_{map1e+p}$	**3.091**	**7.544**	**2.759**
3D-CODED	8.553	10.463	7.058
PointNet++	26.837	81.379	18.23

Table 6. Mean squared reconstruction losses on the DFAUST testset with 5% of the shape bounding box gaussian noise. We show the edge length reconstruction loss (EL), the Point cloud rotation invariant reconstruction loss (PC) and the per triangle area difference

	EL (10^{-5})	PC (10^{-4})	area (10^{-8})
Shape VAE	3.415	**3.393**	2.780
Ours $L_{map1e+p}$	**1.844**	3.400	**1.685**
3D-CODED	6.219	6.898	5.341
PointNet++	36.223	117.824	27.541

Table 7. Mean squared reconstruction losses on the DFAUST testset with under-sampled (500 points)point clouds. Edge length reconstruction loss (EL), Point cloud rotation invariant reconstruction loss (PC) and per triangle area difference are showed.

	EL	PC	area
Shape VAE	$3.492{*}10^{-5}$	$3.774{*}10^{-4}$	$3.332{*}10^{-8}$
Ours $L_{map1e+p}$	$1.908{*}10^{-5}$	$3.373{*}10^{-4}$	$1.747{*}10^{-8}$
Ours L_{direct}	0.1019	0.6289	$1.338{*}10^{-2}$

Table 8. Mean squared reconstruction losses on the DFAUST testset.

B. Ablation study

Importance of cycle consistency loss. We train the mapping networks with direct reconstruction losses instead of cycle consistency losses as described in Eq. (8).

$$\begin{aligned} L_{direct}(P,E) = \alpha\|\text{dec}_p(N_{e2p}(\text{enc}_e(E)))) - P\|^2 \\ + \beta\|el(\text{dec}_p(N_{e2p}(\text{enc}_e(E))))) - E\|^2 \\ + \|\text{dec}_e(N_{p2e}(\text{enc}_p(P))) - E\|^2 \end{aligned} \quad (8)$$

In Table 8, we observe that the quality of the map and the quality of the reconstructions are less high. In Figure 10 we show the cumulative distribution function of the edge length reconstruction loss on the testset. While most shapes seem to have reasonable edge reconstruction quality, outlier points make the reconstruction loss explode. Since cycle consistency is not enforced, the network can map shapes onto outliers in the shape space that do not correspond to reasonable natural shapes.

Effect of training networks separately In our experiments, we fix the weights of the shape VAE and edge autoencoder during the training of the mapping networks. By doing so, we fix the latent space and generating capabilities of each network. We believe that if this constraint is not

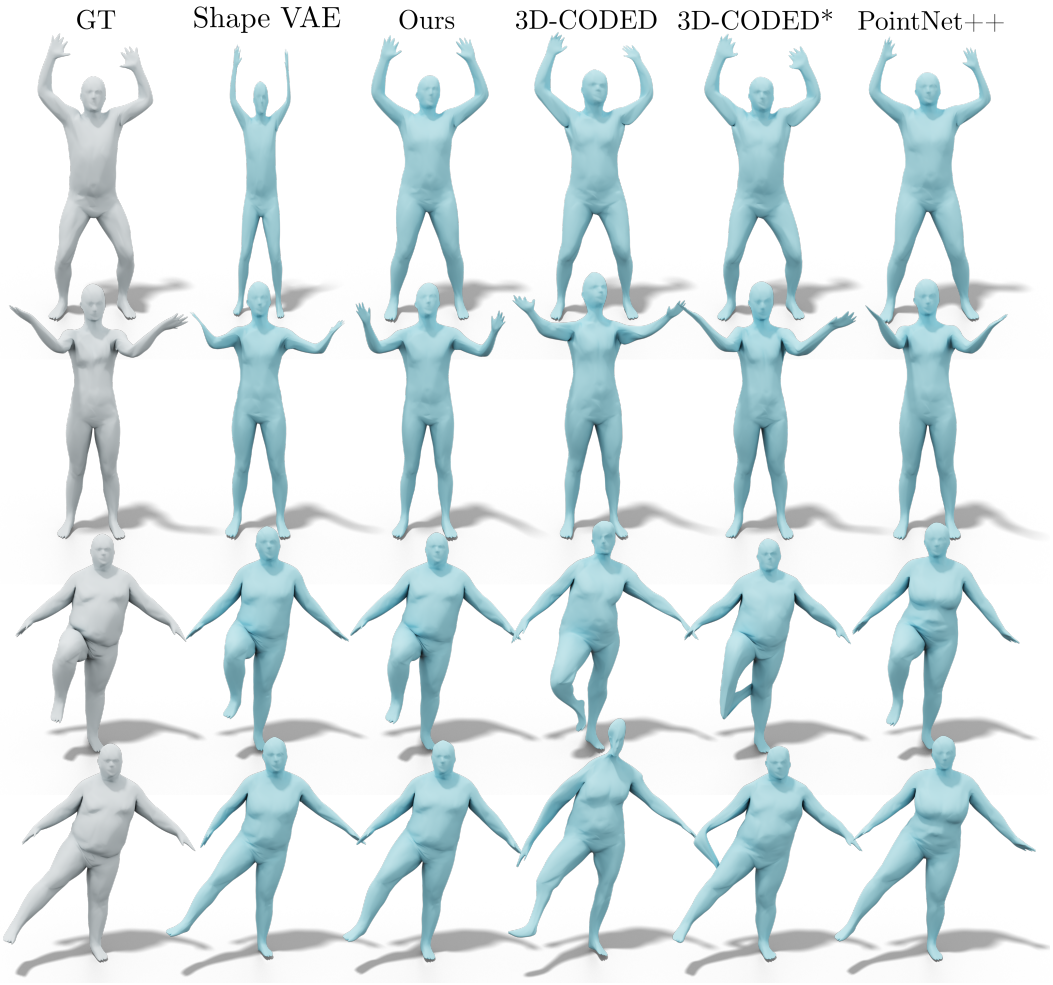

Figure 7. Reconstruction of meshes from 1000 clean vertices of the input shape mesh.

respected, the shape VAE and edge auto-encoder can be indirectly trained for different losses and generate distortions in the generated shapes. Here, we train the mapping networks, edge auto-encoder and shape VAE at the same time. To make the training easier, we use a pretrained shape VAE and edge auto-encoder. As seen in Table 9, the reconstruction losses are better than before. However, the shape VAE can produce non natural reconstructions during interpolations as shown in Figure 11. We believe that if the shape VAE and edge auto-encoder network were not pretrained, the resulting reconstructed shapes would present even more distortions since the pretrained shape VAE can already generate decent natural looking shapes on parts of the dataset.

Linearity regularization term in edge auto-encoder. We train a version of our network without the linearity regularization term L_{lin} described in Eq. (3) of the main manuscript for training the edge auto-encoder. As seen in Table 10, the interpolations in the latent space of the edge auto-encoder are smoother when the network is trained with the linearity term. In Table 11, we observe that this term is also related to smoother interpolations of shapes.

	EL (10^{-5})	PC (10^{-4})	area (10^{-8})
Ours $L_{maple+p}$	1.666	2.611	1.554
Ours sim. train.	1.027	1.464	1.027

Table 9. Mean squared reconstruction losses on the DFAUST test-set. We present our main network and an alternative model where all three components are trained simultaneously. Edge length reconstruction loss (EL), Point cloud rotation invariant reconstruction loss (PC) and per triangle area difference (area).

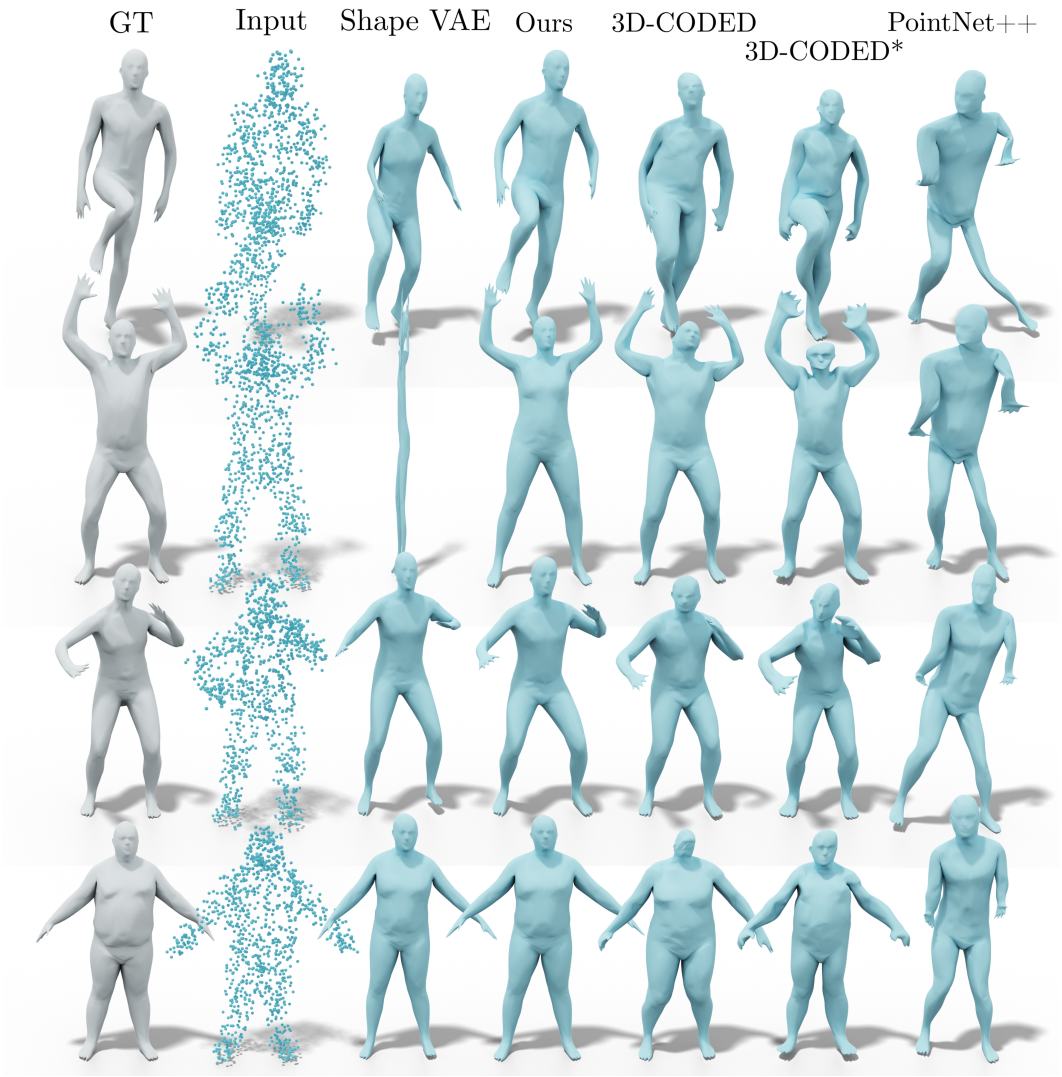

Figure 8. Reconstructions from point clouds with 5% of the shape scale gaussian noise.

	EL
Edge AE $L_{map1e+p}$	0.199
Edge AE no lin. reg.	1.777

Table 10. We report the mean squared variance of the edge length (EL) over the interpolation in the edge length AE latent space of 100 shape pairs.

	EL	area (10^{-4})	volume (10^{-4})
Ours $L_{map1e+p}$	0.230	1.220	0.385
Ours no lin. reg.	0.245	1.361	0.430

Table 11. Reconstruction losses for our network where the edge auto-encoder is trained with and without linearity regularization term. We report the mean squared variance of the edge length (EL), per surface area and total shape volume over the interpolations of 100 shape pairs from the DFAUST testset.

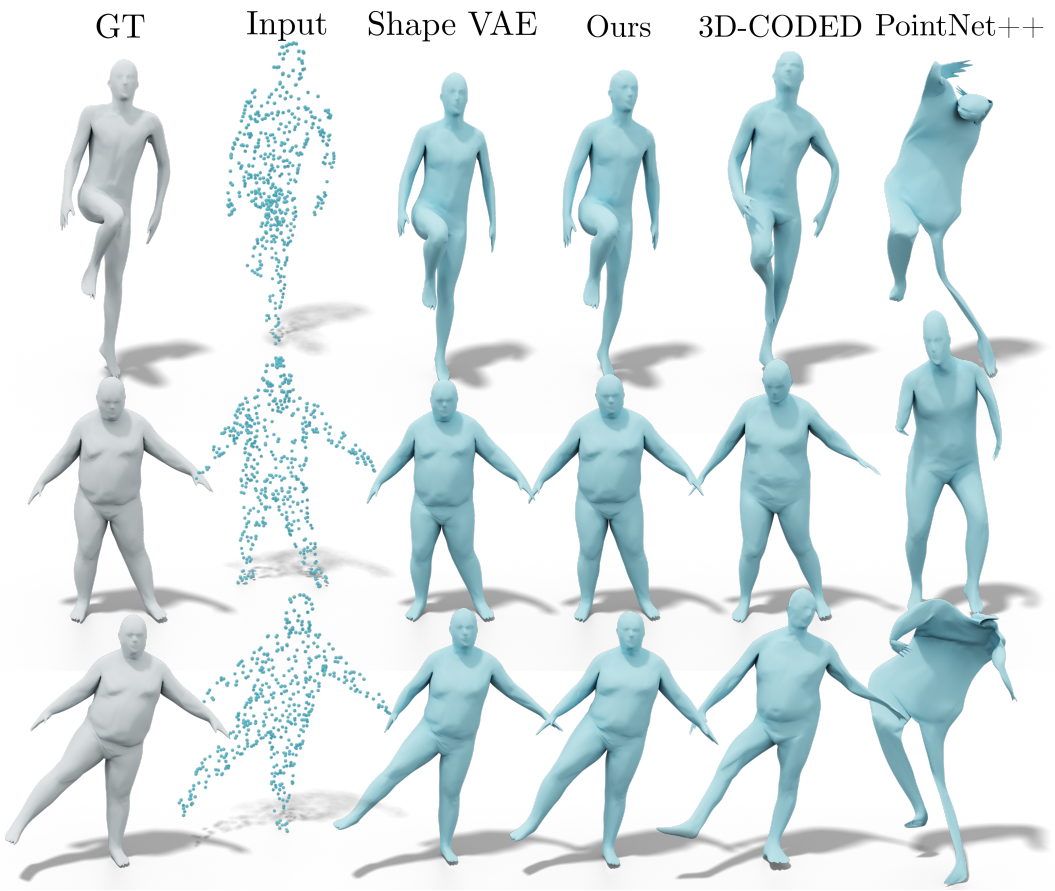

Figure 9. We reconstruct a mesh from 500 points sub-sampled randomly from the ground truth mesh. We use a network pre-trained on inputs of size 1000 points.

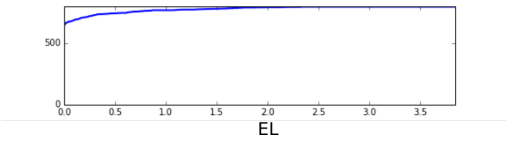

Figure 10. Cumulative distribution function of edge reconstruction loss on the DFAUST testset for our network trained without cycle consistency with L_{direct}.

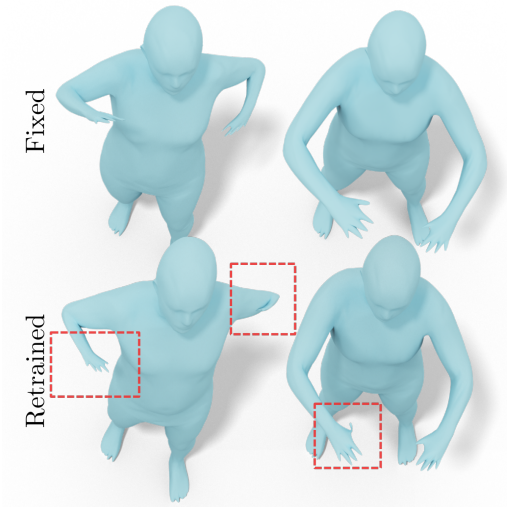

Figure 11. Shape distortions are appearing during interpolation if the shape VAE, edge auto-encoder and mapping networks are trained at the same time.