

Geometric Disentanglement for Generative Latent Shape Models

Tristan Aumentado-Armstrong*, Stavros Tsogkas, Allan Jepson, Sven Dickinson
 University of Toronto Vector Institute for AI Samsung AI Center, Toronto
 taumen@cs.utoronto.ca, {stavros.t,allan.jepson,s.dickinson}@samsung.com

Abstract

Representing 3D shape is a fundamental problem in artificial intelligence, which has numerous applications within computer vision and graphics. One avenue that has recently begun to be explored is the use of latent representations of generative models. However, it remains an open problem to learn a generative model of shape that is interpretable and easily manipulated, particularly in the absence of supervised labels. In this paper, we propose an unsupervised approach to partitioning the latent space of a variational autoencoder for 3D point clouds in a natural way, using only geometric information. Our method makes use of tools from spectral differential geometry to separate intrinsic and extrinsic shape information, and then considers several hierarchical disentanglement penalties for dividing the latent space in this manner, including a novel one that penalizes the Jacobian of the latent representation of the decoded output with respect to the latent encoding. We show that the resulting representation exhibits intuitive and interpretable behavior, enabling tasks such as pose transfer and pose-aware shape retrieval that cannot easily be performed by models with an entangled representation.

1. Introduction

Fitting and manipulating 3D shape (e.g., for inferring 3D structure from images or efficiently computing animations) are core problems in computer vision and graphics. Unfortunately, designing an appropriate representation of 3D object shape is a non-trivial, and, often, task-dependent issue.

One way to approach this problem is to use deep generative models, such as generative adversarial networks (GANs) [19] or variational autoencoders (VAEs) [54, 34]. These methods are not only capable of generating novel examples of data points, but also produce a latent space that

*The work in this article was done while Tristan A.A. was a student at the University of Toronto. Sven Dickinson, Allan Jepson, and Stavros Tsogkas contributed in their capacity as Professors and Postdoc at the University of Toronto, respectively. The views expressed (or the conclusions reached) are their own and do not necessarily represent the views of Samsung Research America, Inc.

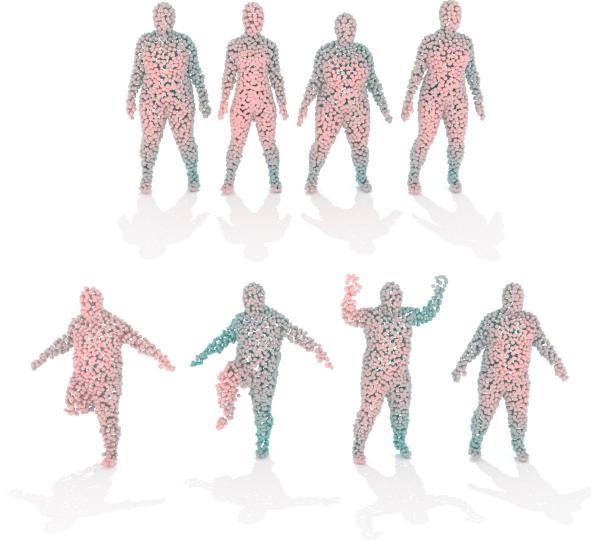


Figure 1. Factoring pose and intrinsic shape within a disentangled latent space offers fine-grained control when generating shapes using a generative model. **Top:** decoded shapes with constant latent extrinsic group and randomly sampled latent intrinsics. **Bottom:** decoded shapes with fixed latent intrinsic group and random extrinsics. Colors denote depth (i.e., distance from the camera).

provides a compressed, continuous vector representation of the data, allowing efficient manipulation. Rather than performing explicit physical calculations, for example, one can imagine performing approximate “intuitive” physics by predicting movements in the latent space instead.

However, a natural representation for 3D objects is likely to be highly structured, with different variables controlling separate aspects of an object. In general, this notion of *disentanglement* [7] is a major tenet of representation learning, that closely aligns with human reasoning, and is supported by neuroscientific findings [5, 27, 25]. Given the utility of disentangled representations, a natural question is whether we can structure the latent space in a purely unsupervised manner. In the context of 3D shapes, this is equivalent to asking how one can factor the representation into interpretable components using geometric information alone.

We take two main steps in this direction. First, we lever-

age methods from spectral differential geometry, defining a notion of intrinsic shape based on the Laplace-Beltrami operator (LBO) spectrum. This provides a fully unsupervised descriptor of shape that can be computed from the geometry alone and is invariant to isometric pose changes. Furthermore, unlike semantic labels, the spectrum is continuous, catering to the intuition that “shape” should be a smoothly deformable object property. It also automatically divorces the intrinsic or “core” shape representation from rigid or isometric (e.g., articulated) transforms, which we call extrinsic shape. Second, we build on a two-level architecture for generative point cloud models [1] and examine several approaches to hierarchical latent disentanglement. In addition to a previously used information-theoretic penalty based on total correlation, we describe a hierarchical flavor of a covariance-based technique, and propose a novel penalty term, based on the Jacobian between latent variables. Together, these methods allow us to learn a factored representation of 3D shape using only geometric information in an unsupervised manner. This representation can then be applied to several tasks, including non-rigid pose manipulation (as in Figure 1) and pose-aware shape retrieval, in addition to generative sampling of new shapes.

2. Related Work

2.1. Latent Disentanglement in Generative Models

A number of techniques for disentangling VAEs have recently arisen, often based on the distributional properties of the latent prior. One such method is the β -VAE [26, 10], in which one can enforce greater disentanglement at the cost of poorer reconstruction quality. As a result, researchers have proposed several information-theoretic approaches that utilize a penalty on the total correlation (TC), a multivariate generalization of the mutual information [61]. Minimizing TC corresponds to minimizing the information shared among variables, making it a powerful disentanglement technique [18, 11, 32]. Yet, such methods do not consider *groups* of latent variables, and do not control the strength of disentanglement *between* versus *within* groups. Since geometric shape properties in our model cannot be described with a single variable, our intrinsic-extrinsic factorization requires *hierarchical* disentanglement. Fortunately, a multi-level decomposition of the ELBO can be used to obtain a hierarchical TC penalty [15].

Other examples of disentanglement algorithms include information-theoretic methods in GANs [12], latent whitening [23], covariance penalization [35], and Bayesian hyperpriors [2]. A number of techniques also utilize known groupings or discrete labels of the data [28, 8, 55, 22]. In contrast, our work does not have access to discrete groupings (given the continuity of the spectrum), requires a hierarchical structuring, and utilizes no domain knowledge out-

side of the geometry itself. We therefore consider three approaches to hierarchical disentanglement: (i) a TC penalty; (ii) a decomposed covariance loss; and (iii) shrinking the Jacobian between latent groups.

2.2. Deep Generative Models of 3D Point Clouds

Point clouds represent a practical alternative to voxel and mesh representations for 3D shape. Although they do not model the complex connectivity information of meshes, point clouds can still capture high resolution details at lower computational cost than voxel-based methods. One other benefit is that much real-world data in computer vision is captured as point sets, which has resulted in considerable effort on learning from point cloud data. However, complications arise from the set-valued nature of each datum [52]. PointNet [50] handles that by using a series of 1D convolutions and affine transforms, followed by pooling and fully-connected layers. Many approaches have tried to integrate neighborhood information into this encoder (e.g., [51, 24, 62, 3]), but this remains an open problem.

Several generative models of point clouds exist: Nash and Williams [45] utilize a VAE on data of 3D part segmentations and associated normals, whereas Achlioptas et al. [1] use a GAN. Li et al. [38] adopt a hierarchical sampling approach with a more general GAN loss, while Valsesia et al. [59] utilize a graph convolutional method with a GAN loss. In comparison to these methods, we focus on unsupervised geometric disentanglement of the latent representation, allowing us to factor pose and intrinsic shape, and use it for downstream tasks. We also do not require additional information, such as part segmentations. Compared to standard GANs, the use of a VAE permits natural probabilistic approaches to hierarchical disentanglement, as well as the presence of an encoder, which is necessary for latent representation manipulations and tasks such as retrieval. In this sense, our work is orthogonal to GAN-based representation learning, and both techniques may be mutually applicable as joint VAE-GAN models advance (e.g., [42, 64]).

Two recent related works utilize meshes for deformation-aware 3D generative modelling. Tan et al. [56] utilize latent manipulation to perform a variety of tasks, but does not explicitly separate pose and shape. Gao et al. [17] fix two domains per model, making intrinsic shape variation and comparing latent vectors difficult. Both works are limited by the need for identical connectivity. In contrast, we can smoothly explore latent shape and pose independently, without labels or correspondence. We further note that our disentanglement framework is modality-agnostic to the extent that only the AE details need change.

In this work, we utilize point cloud data to learn a latent representation of 3D shape, capable of encoding, decoding, and novel sampling. Using PointNet as the encoder, we define a VAE on the latent space of a deterministic autoen-

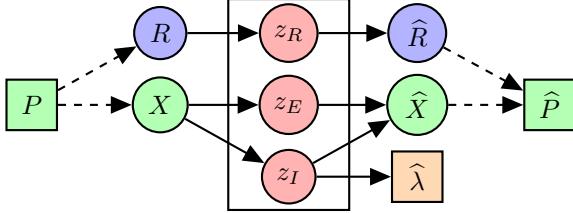


Figure 2. A schematic overview of the combined two-level architecture used as the generative model. A point cloud P is first encoded into (R, X) by a *deterministic* AE based on PointNet, R being the quaternion representing the rotation of the shape, and X the compressed representation of the input shape. (R, X) is then further compressed into a latent representation $z = (z_R, z_E, z_I)$ of a VAE. The hierarchical latent variable z has disentangled subgroups in red (representing rotation, extrinsics, and intrinsics, respectively). The intrinsic latent subgroup z_I is used to predict the LBO spectrum $\hat{\lambda}$. Both the extrinsic z_E and intrinsic z_I are utilized to compute the shape \hat{X} in the AE’s latent space. The latent rotation z_R is used to predict the quaternion \hat{R} . Finally, the decoded representation (\hat{R}, \hat{X}) is used to reconstruct the original point cloud \hat{P} . The deterministic AE mappings are shown as dashed lines; VAE mappings are represented by solid lines.

coder, similar to [1]. Our main goal is to investigate how unsupervised geometric disentanglement using spectral information can be used to structure the latent space of shape in a more interpretable and potentially more useful manner.

3. Point Cloud Autoencoder

Similar to prior work [1], we utilize a two-level architecture, where the VAE is learned on the latent space of an AE. This architecture is shown in Figure 2. Throughout this work, we use the following notation: P denotes a point cloud, (R, X) is the latent AE representation, and \hat{P} is the reconstructed point cloud. Although rotation is a strictly extrinsic transformation, we separate them because (1) rotation is intuitively different than other forms of non-rigid extrinsic pose (e.g., articulation), (2) having separate control over rotations is commonly desirable in applications (e.g., [31, 16]), and (3) our quaternion-based factorization provides a straightforward way to do so.

3.1. Point Cloud Losses

Following previous work on point cloud AEs [1, 39, 14], we utilize a form of the Chamfer distance as our main measure of similarity. We define the max-average function

$$M_\alpha(\ell_1, \ell_2) = \alpha \max\{\ell_1, \ell_2\} + (1 - \alpha)(\ell_1 + \ell_2)/2, \quad (1)$$

where α is a hyper-parameter that controls the relative weight of the two values. It is useful to weight the larger of the two terms higher, so that the network does not focus

on only one term [63]. We then use the point cloud loss

$$\mathcal{L}_C = M_{\alpha_C} \left(\frac{1}{|P|} \sum_{p \in P} \hat{d}(p), \frac{1}{|\hat{P}|} \sum_{\hat{p} \in \hat{P}} d(\hat{p}) \right), \quad (2)$$

where $d(\hat{p}) = \min_{p \in P} \|p - \hat{p}\|_2^2$ and $\hat{d}(p) = \min_{\hat{p} \in \hat{P}} \|p - \hat{p}\|_2^2$. In an effort to reduce outliers, we add a second term, as a form of approximate Hausdorff loss:

$$\mathcal{L}_H = M_{\alpha_H} \left(\max_{p \in P} d(\hat{p}), \max_{\hat{p} \in \hat{P}} \hat{d}(p) \right). \quad (3)$$

The final reconstruction loss is therefore $\mathcal{L}_R = r_C \mathcal{L}_C + r_H \mathcal{L}_H$ for constants r_C, r_H .

3.2. Quaternionic Rotation Representation

We make use of quaternions to represent rotation in the AE model. The unit quaternions form a double cover of the rotation group $SO(3)$ [29]; hence, any vector $R \in \mathbb{R}^4$ can be converted to a rotation via normalization. We can then differentiably convert any such quaternion R to a rotation matrix R_M . To take the topology of $SO(3)$ into account, we use the distance metric [29] $\mathcal{L}_Q = 1 - |q \cdot \tilde{q}|$ between unit quaternions q and \tilde{q} .

3.3. Autoencoder Model

The encoding function $f_E(P) = (R, X)$ maps a point cloud P to a vector $(R, X) \in \mathbb{R}^{D_A}$, which is partitioned into a quaternion R (representing the rotation) and a vector X , which is a compressed representation of the shape. The mapping is performed by a PointNet model [50], followed by fully connected (FC) layers. The decoding function works by rotating the decoded shape vector: $f_D(R, X) = g_D(X)R_M = \hat{P}$, where g_D was implemented via FC layers and R_M is the matrix form of R . The loss function for the autoencoder is the reconstruction loss \mathcal{L}_R .

Note that the input can be a point cloud of arbitrary size, but the output is of fixed size, and is determined by the final network layer (though alternative architectures could be dropped in to avoid this limitation [38, 20]). Our data augmentation scheme during training consists of random rotations of the data about the height axis, and using randomly sampled points from the shape as input (see Section 5). For architectural details, see Supplementary Material.

4. Geometrically Disentangled VAE

Our generative model, the geometrically disentangled VAE (GDVAE), is defined on top of the latent space of the AE; in other words, it encodes and decodes between its own latent space (denoted z) and that of the AE (i.e., (R, X)). The latent space of the VAE is represented by a vector that is hierarchically decomposed into sub-parts, $z = (z_R, z_E, z_I)$,

representing the rotational, extrinsic, and intrinsic components, respectively. In addition to reconstruction loss, we define the following loss terms: (1) a probabilistic loss that matches the latent encoder distribution to the prior $p(z)$, (2) a spectral loss, which trains a network to map z_I to a spectrum λ , and (3) a disentanglement loss that penalizes the sharing of information between z_I and z_E in the latent space. Note that the first (1) and third (3) terms are based on the Hierarchically Factorized VAE (HFVAE) defined by Esmaeili et al. [15], but the third term also includes a covariance penalty motivated by the Disentangled Inferred Prior VAE (DIP-VAE) [35] and another penalty based on the Jacobian between latent subgroups. In the next sections, we discuss each term in more detail.

4.1. Latent Disentanglement Penalties

To disentangle intrinsic and extrinsic geometry in the latent space, we consider three different hierarchical penalties. In this section, we define the latent space z to consist of $|G|$ subgroups, i.e., $z = (z_1, \dots, z_{|G|})$, with each subset z_i being a vector-valued variable of length g_i . We wish to disentangle each subgroup from all the others. In this work, $z = (z_R, z_E, z_I)$ and $|G| = 3$.

Hierarchically Factorized Variational Autoencoder. Recent work by Esmaeili et al. [15] showed that the prior-matching term of the VAE objective (i.e., $\mathcal{D}_{\text{KL}}[q_\phi(z|x) \parallel p(z)]$) can be hierarchically decomposed as

$$\mathcal{L}_{\text{HF}} = \beta_1 P_{\text{intra}} + \beta_2 P_{\text{KL}} + \beta_3 \mathcal{I}[x; z] + \beta_4 \text{TC}(z), \quad (4)$$

where $\text{TC}(z)$ is the inter-group TC, $\mathcal{I}[x; z]$ is the mutual information between the data and its latent representation, and P_{intra} and P_{KL} are the intra-group TC and dimension-wise KL-divergence, respectively, given by the following formulas: $P_{\text{intra}} = \sum_g \text{TC}(z_g)$ and $P_{\text{KL}} = \sum_{g,d} \mathcal{D}_{\text{KL}}[q_\phi(z_{g,d}) \parallel p(z_{g,d})]$.

As far as disentanglement is concerned, the main term enforcing inter-group independence (via the TC) is the one weighted by β_4 . However, note that the other terms are essential for matching the latent distribution to the prior $p(z)$, which allows generative sampling from the network. We use the implementation in ProbTorch [44].

Hierarchical Covariance Penalty. A straightforward measure of statistical dependence is covariance. While this is only a measure of the linear dependence between variables, unlike the information-theoretic penalty considered above, vanishing covariance is still necessary for disentanglement. Hence, we consider a covariance-based penalty to enforce independence between variable groups. This is motivated by Kumar et al. [35], who discuss how disentanglement can be better controlled by introducing a penalty

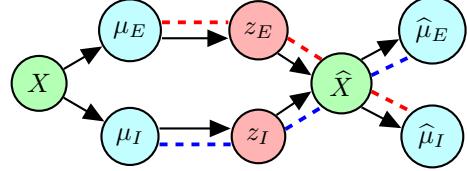


Figure 3. Diagram of the pairwise Jacobian norm penalty computation within a VAE. The red and blue dashed paths show the computation graph paths utilized to compute the Jacobians.

that moment-matches the inferred prior $q_\phi(z)$ to the latent prior $p(z)$. We perform a simple alteration to make this penalty hierarchical. Specifically, let \widehat{C} denote the estimated covariance matrix over the batch and recall that $q_\phi(z|x) = \mathcal{N}(z|\mu_\phi(x), \Sigma_\phi(x))$. Finally, denote μ_g as the part of $\mu_\phi(x)$ corresponding to group g (i.e., parameterizing the approximate posterior over z_g) and define

$$\mathcal{L}_{\text{cov}} = \gamma_I \sum_{g \neq \tilde{g}} \sum_{i,j} \left| \widehat{C}(\mu_g, \mu_{\tilde{g}})_{ij} \right| \quad (5)$$

as a penalty on inter-group covariance, where the first sum is taken over all non-identical pairings. We ignore the additional moment-matching penalties on the diagonal and intra-group covariance from [35], since they are not related to intrinsic-extrinsic disentanglement and a prior-matching term is already present within \mathcal{L}_{HF} .

Pairwise Jacobian Norm Penalty. Finally, we follow the intuition that changing the value of one latent group should not affect the expected value of any other group. We derive a loss term for this by considering how the variables change if the decoded shape is re-encoded into the latent space. This approach to geometric disentanglement is visualized in Figure 3. Unlike the TC and covariance-based penalties, this does not disentangle z_R from z_E and z_I .

Formally, we consider the Jacobian of a latent group with respect to another. The norm of this Jacobian can be viewed as a measure of how much one latent group can affect another group, through the decoder. This measure is

$$\mathcal{L}_J = \max_{g \neq \tilde{g}} \left\| \frac{\partial \widehat{\mu}_g}{\partial \mu_{\tilde{g}}} \right\|_F^2, \quad (6)$$

where \widehat{X} is the decoded shape, $\widehat{\mu}_g$ represents group g from $\mu_\phi(\widehat{X})$, and we take the maximum over pairs of groups.

4.2. Spectral Loss

Mathematically, the intrinsic differential geometry of a shape can be viewed as those properties dependent only on the metric tensor, i.e., independent of the embedding of the shape [13]. Such properties depend only on geodesic distances on the shape rather than how the shape sits in the

ambient 3D space. The Laplace-Beltrami operator (LBO) is a popular way of capturing intrinsic shape. Its spectrum λ can be formally described by viewing a shape as a 2D Riemannian manifold (\mathcal{M}, g) embedded in 3D, with point clouds being viewed as random samplings from this surface.

Given the spectrum λ of a shape, we wish to compute a loss with respect to a predicted spectrum $\hat{\lambda}$, treating each as a vector with N_λ elements. The LBO spectrum has a very specific structure, with $\lambda_i \geq 0 \forall i$ and $\lambda_j \geq \lambda_k \forall j > k$. Analogous to frequency-space signal processing, larger elements of λ correspond to “higher frequency” properties of the shape itself: i.e., finer geometric details, as opposed to coarse overall shape. This analogy can be formalized by the “manifold harmonic transform”, a direct generalization of the Fourier transform to non-Euclidean domains based on the LBO [58]. Due to this structure, a naive vector space loss function on λ (e.g., L_2) will over-weight learning the higher frequency elements of the spectrum. We suggest that the lower portions of λ not be down-weighted, as they are less susceptible to noise and convey larger-scale, “low-frequency” global information about the shape, which is more useful for coarser shape reconstruction.

Given this, we design a loss function that avoids over-weighting the higher frequency end of the spectrum:

$$\mathcal{L}_S(\lambda, \hat{\lambda}) = \frac{1}{N_\lambda} \sum_{i=1}^{N_\lambda} \frac{|\lambda_i - \hat{\lambda}_i|}{i}, \quad (7)$$

where the use of the L_1 norm and the linearly increasing element-wise weight of i decrease the disproportionate effect of the larger magnitudes at the higher end of the spectrum. The use of linear weights is theoretically motivated by Weyl’s law (e.g., [53]), which asserts that spectrum elements increase approximately linearly, for large i .

4.3. VAE Model

Essentially, the latent space is divided into three parts, for rotational, extrinsic, and intrinsic geometry, denoted z_R , z_E , and z_I , respectively. We note that, while rotation is fundamentally extrinsic, we can take advantage of the AE’s decomposed representation to define z_R on the AE latent space over R , and use z_E and z_I for X . The encoder model can be written as $(z_E, z_I) = \mu_\phi(X) + \Sigma_\phi(X)\xi$, where $\xi \sim \mathcal{N}(0, I)$, while the decoder is written $\hat{X} = h_D(z_E, z_I)$. A separate encoder-decoder pair is used for R . The spectrum is predicted from the latent intrinsics alone: $\hat{\lambda} = f_S(z_I)$.

The reconstruction loss, used to compute the log-likelihood, is given by the combination of the quaternion metric and a Euclidean loss between the vector representation of the (compressed) shape and its reconstruction:

$$\mathcal{L}_V = \frac{1}{D} \|X - \hat{X}\|_2^2 + w_Q \mathcal{L}_Q, \quad (8)$$

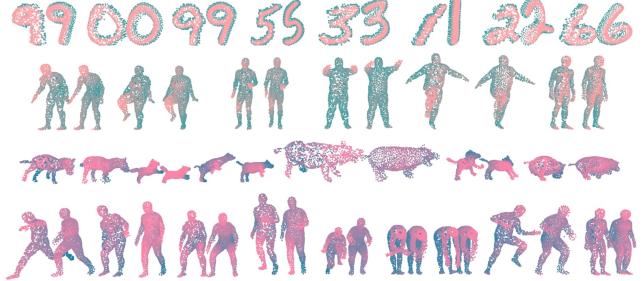


Figure 4. **Reconstructions** of random samples, passed through both the AE and VAE. For each pair, the left shape is the input and the right shape is the reconstruction. Colors denote depth (i.e., distance from the camera). Rows: MNIST, Dyna, SMAL, SMPL.

where \mathcal{L}_Q is the metric over quaternion rotations and $D = \dim(X)$. We now define the overall VAE loss:

$$\mathcal{L} = \eta \mathcal{L}_V + \mathcal{L}_{HF} + \mathcal{L}_{COV} + w_J \mathcal{L}_J + \zeta \mathcal{L}_S. \quad (9)$$

The VAE needs to be able to (1) autoencode shapes, (2) sample novel shapes, and (3) disentangle latent groups. The first term of \mathcal{L} encourages (1), while the second term enables (2); the last four terms of \mathcal{L} contribute to task (3).

5. Experiments

For our experiments, we consider four datasets of meshes: shapes computed from the MNIST dataset [37], the MPI Dyna dataset of human shapes [49], a dataset of animal shapes from the Skinned Multi-Animal Linear model (SMAL) [65], and a dataset of human shapes from the Skinned Multi-Person Linear model (SMPL) [40] via the SURREAL dataset [60]. For each, we generate point clouds of size N_T via area-weighted sampling.

For SMAL and SMPL we generate data from 3D models using a modified version of the approach in Groueix et al. [20]. During training, the input of the network is a uniformly random subset of N_S points from the original point cloud. We defer to the Supplemental Material for details concerning dataset processing and generation.

We compute the LBO spectra directly from the triangular meshes using the cotangent weights formulation [43], as it provides a more reliable result than algorithms utilizing point clouds (e.g., [6]). We thus obtain a spectrum λ as a N_λ -dimensional vector, associated with each shape. We note that our algorithm requires only a point cloud as input data (or a Gaussian random vector, if generating samples). LBO spectra are utilized only at training time, while triangle meshes are used only for training set generation. Hence, our method remains applicable to pure point cloud data.

5.1. Generative Shape Modeling

Ideally, our model should be able to disentangle intrinsic and extrinsic geometry without losing its capacity to (1) re-

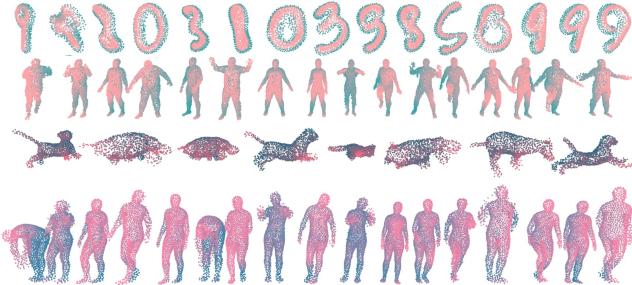


Figure 5. **Samples drawn** from the latent space of the VAE by decoding $z \sim \mathcal{N}(0, I)$ with $z_R = 0$. Colors denote depth (i.e., distance from the camera). Rows: MNIST, Dyna, SMAL, SMPL.

z_R	z_E	z_I	z_{RE}	z_{RI}	z_{EI}	z	S
0.32	0.47	0.60	0.64	0.68	0.88	0.88	0.98

Table 1. Accuracies of a linear classifier on various segments of the latent space from the MNIST test set. We denote $z_{RE} = (z_R, z_E)$, $z_{RI} = (z_R, z_I)$, $z_{EI} = (z_E, z_I)$, and $S = (R, X)$.

construct point clouds and (2) generate random shape samples. We show qualitative reconstruction results in Figure 4. Considering the latent dimensionalities ($|z_E|, |z_I|$ are 5, 5; 10, 10; 8, 5; and 12, 5, for MNIST, Dyna, SMAL, and SMPL, respectively), it is clear that the model is capable of decoding from significant compression. However, thin or protruding areas (e.g., hands or legs) have a lower point density (a known problem with the Chamfer distance [1]).

We also consider the capacity of the model to generate novel shapes from randomly sampled latent z values, as shown in Figure 5. We can see a diversity of shapes and poses; however, not all samples belong to the data distribution (e.g., invalid MNIST samples, or extra protrusions from human shapes). VAEs are known to generate blurry images [57, 36]; in our case, ‘‘blurring’’ implies a perturbation in the latent space, rather than in the 3D point positions, explaining the unintuitive artifacts in Figures 4 and 5.

A standard evaluation method in generative modeling is testing the usefulness of the representation in downstream tasks (e.g., [1]). This is also useful for illustrating the role of the latent disentanglement. As such, we utilize our encodings for classification on MNIST, recalling that our representation was learned without access to the labels. To do so, we train a linear support vector classifier (from scikit-learn [47], with default parameters and no data augmentation) on the parts of the latent space defined by the GDVAE (see Table 1). Comparing the drop from $S = (R, X)$ to z shows the effect of compression and KL regularization; we can also see that z_R is the least useful component, but that it still performs better than chance, suggesting a correlation between digit identity and the orientation encoded by the network. In the Supplemental Material, we include confusion matrices showing that mistakes on z_I or (z_R, z_I) are similar to those incurred when using λ directly.

Lastly, our AE naturally disentangles rigid pose (rotation) and the rest of the representation. Ideally, the network would not learn disparate X representations for a single shape under rotation; rather, it should map them to the same shape representation, with a different accompanying quaternion. This would allow rigid pose normalization via derotations: for instance, rigid alignment of shapes could be done by matching z_R , which could be useful for pose normalizing 3D data. We found that the model is robust to small rotations, but it often learns separate representations under larger rotations (see Supplemental Material). In some cases, this may be unavoidable (e.g., for MNIST, 9 and 6 are often indistinguishable after a 180° rotation).

5.2. Disentangled Latent Shape Manipulation

We provide a qualitative examination of the properties of the geometrically disentangled latent space. For human and animal shapes, we expect z_E to control the articulated pose, while z_I should independently control the intrinsic body shape. We show the effect of traversing the latent space within its intrinsic and extrinsic components separately, via linear interpolations between shapes in Figure 6 (fixing $z_R = 0$). We observe that moving in z_I (horizontally) largely changes the body type of the subject, associated with identity in humans or species among animals, whereas moving in z_E (vertically) mostly controls the articulated pose. Moving in the diagonal of each inset is akin to latent interpolation in a non-disentangled representation.

We can also consider the viability of our method for pose transfer, by transferring latent extrinsics between two shapes. Although the analogous pose is often exchanged (see Figure 7), there are some failure cases: for example, on SMPL and Dyna, the transferred arm positions tend to be similar, but not exactly the same. This suggests a failure in the disentanglement, since the articulations are tied to the latent intrinsics z_I . In general, we found that latent manipulations starting from real data (e.g., interpolations or pose transfers between real point clouds) gave more interpretable results than those from latent samples, suggesting the model sometimes struggled to match the approximate posterior to the prior, particularly for the richer datasets from SMAL and SMPL. Nevertheless, on the Dyna set, we show that randomly sampling z_E or z_I can still give intuitive alterations to pose versus intrinsic shape (Figure 8).

5.3. Pose-Aware Shape Retrieval

We next apply our model to a classical computer vision task: 3D shape retrieval. Note that our disentangled representation also affords retrieving shapes based exclusively on intrinsic shape (ignoring isometries) or articulated pose (ignoring intrinsics). While the former can be done via spectral methods (e.g., [9, 48]), the latter is less straightforward. Our method also works directly on raw point clouds.

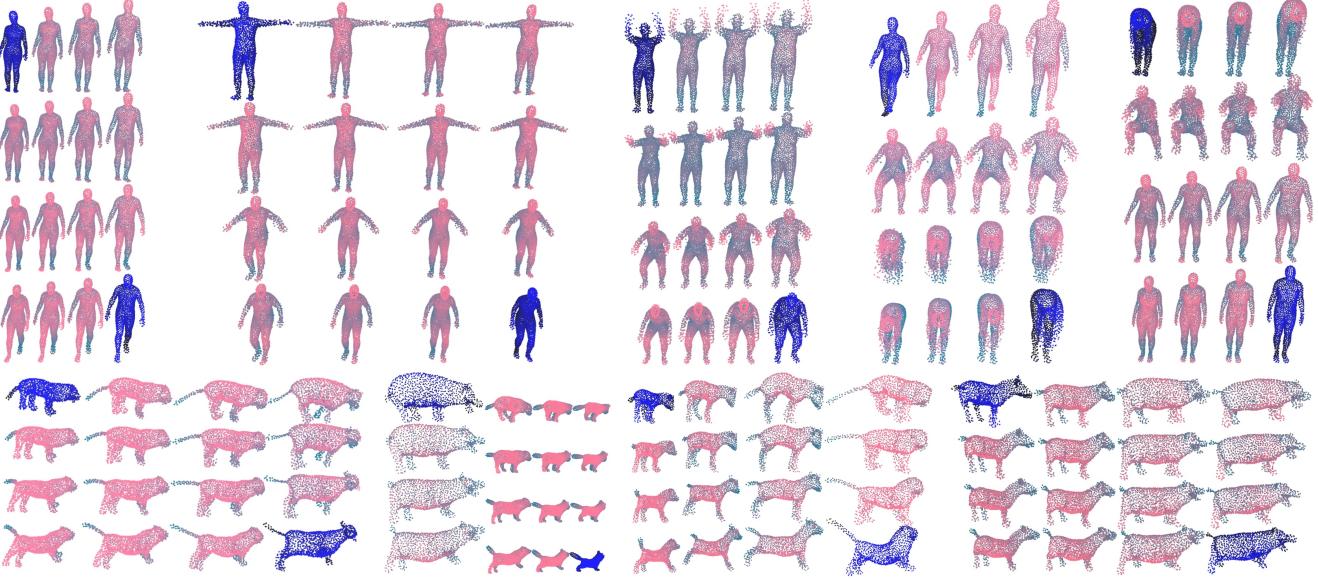


Figure 6. **Latent space interpolations** between SMPL (row 1) and SMAL (row 2) shapes. Each inset interpolates z between the upper-left and lower right shapes, with z_E changing along the vertical axis and z_I changing along the horizontal one. Per-shape colours denote depth.

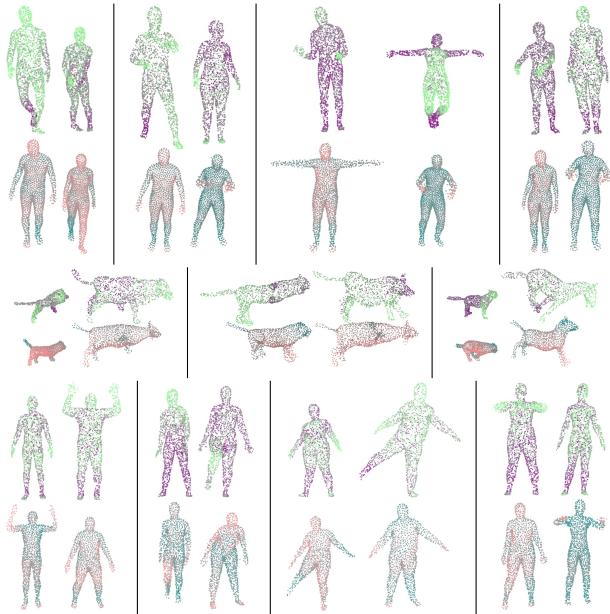


Figure 7. **Pose transfer** via exchanging latent extrinsics. Per inset of four shapes, the bottom shapes have the z_R and z_I of the shape directly above, but the z_E of their diagonally opposite shape in the top row. Per-shape colors denote depth. Upper shapes are real point clouds; lower ones are reconstructions after latent transfer. Rows: SMPL, SMAL, and Dyna examples.

We measure our performance on this task using the synthetic datasets from SMAL and SMPL. Since both are defined by intrinsic shape variables (β) and articulated pose parameters (Rodrigues vectors at joints, θ), we can use knowledge of these to validate our approach quantitatively.

Note that our model only ever sees raw point clouds (i.e., it cannot access β or θ values). Our approach is simple: after training, we encode each shape in a held-out test set, and then use the L_2 distance in the latent spaces (X , z , z_E , and z_I) to retrieve nearest neighbours. We measure the error in terms of how close the β and θ values of the query P_Q (β_Q, θ_Q) are to those of a retrieved shape P_R (β_R, θ_R). We define the distance $E_\beta(P_Q, P_R)$ between the shape intrinsics as the mean squared error $MSE(\beta_Q, \beta_R)$. To measure extrinsic pose error, we first transform the axis-angle representation θ to the equivalent unit quaternion $q(\theta)$, and then compute $E_\theta(P_Q, P_R) = \mathcal{L}_Q(q(\theta_Q), q(\theta_R))$. We also normalize each error by the average error between all shape pairs, thus measuring our performance compared to a uniformly random retrieval algorithm. Ideally, retrieving via z_E should have a high E_β and a low E_θ , while using z_I should have a high E_θ and a low E_β .

Table 2 shows the results. Each error is computed using the mean error over the top three matched shapes per query, averaged across the set. As expected, the E_β for z_I is much lower than for z_E (and z on SMAL), while the E_θ for z_E is much lower than that of z_I (and z on SMPL). Just as importantly, from a disentanglement perspective, we see that the E_β of z_E is much higher than that of z , as is the E_θ of z_I . We emphasize that E_β and E_θ measure different quantities, and should not be directly compared; instead, each error type should be compared across the latent spaces. In this way, z and X serve as non-disentangled baselines, where both error types are low. This provides a quantitative measure of geometric disentanglement which shows that our unsupervised representation is useful for generic tasks, such as



Figure 8. Effect of **randomly sampling either the intrinsic or extrinsic components** of four Dyna shapes. **Leftmost shape:** original input; **upper row:** $z_I \sim \mathcal{N}(0, I)$, fixed z_E ; **lower row:** $z_E \sim \mathcal{N}(0, I)$, fixed z_I . Colors denote depth (distance from the camera).

	X	z	z_E	z_I
SMAL	E_β	0.641	0.743	0.975
	E_θ	0.938	0.983	0.983
SMPL	E_β	0.856	0.922	0.997
	E_θ	0.577	0.726	0.709

Table 2. Error values for retrieval tasks, using various latent representations. Values are averaged over three models trained with the same hyper-parameters, with each model run three times to account for randomness in the point set sampling of the input shapes. (See Supplemental Material for standard errors).



Figure 9. **Shape retrieval.** Per inset: leftmost shape is query, middle two shapes are retrieved via z_E , and rightmost two shapes are retrieved via z_I . Color gradients per shape denote depth.

retrieval. Figure 9 shows some examples of retrieved shapes using z_E and z_I . The high error rates, however, do suggest that there is still much room for improvement.

5.4. Disentanglement Penalty Ablations

We use three disentanglement penalties to control the structure of the latent space, based on the inter-group total correlation (TC), covariance (COV), and Jacobian (J). To discern the contributions of each, we conduct the following experiments (details and figures are in the Supplemental).

We first train several models on MNIST, monitoring the loss curves while we vary the strength of each penalty. We find that higher TC penalties substantially reduce COV and J, while COV and J are less effective in reducing TC. This suggests TC is a “stronger” penalty than COV and J, which is intuitive, given that it directly measures information, rather than linear relationships (as COV does) or local ones (as J does). Nevertheless, it does not remove the entanglement measured in COV and J as effectively as direct

penalties on them, and using higher TC penalties quickly leads to lower reconstruction performance. Using all three penalties achieves the lowest values for all measures.

We then perform a more specific experiment on the SMAL and SMPL datasets, ablating the COV and/or J penalties, and examining both the loss curves and the retrieval results. Particularly on SMPL, the presence of a direct penalty on COV and J is very useful in reducing their respective values. Regarding retrieval, the E_β using z_I on SMAL and the E_θ using z_E on SMPL were lowest using all three penalties. Interestingly, E_β using z_I on SMPL and E_θ using z_E on SMAL could be improved without COV and J; however, such decreases were concomitant with reductions in E_θ using z_I and E_β using z_E , which suggests increased entanglement. While not exhaustive, these experiments suggest the utility of applying all three terms.

We also considered the effect of noise in the spectra estimates (see Supplemental Material). The network tolerates moderate spectral noise, with decreasing disentanglement performance as the noise increases. In practice, one may use meshes with added noise for data augmentation, to help generalization to noisy point clouds at test time.

6. Conclusion

We have defined a novel, two-level unsupervised VAE with a disentangled latent space, using *purely geometric* information (i.e., without semantic labels). We have considered several hierarchical disentanglement losses, including a novel penalty based on the Jacobian of the latent variables of the reconstruction with respect to the original latent groups, and have examined the effects of the various penalties via ablation studies. Our disentangled architecture can effectively compress vector representations via encoding and perform generative sampling of new shapes. Through this factored representation, our model permits several downstream tasks on 3D shapes (such as pose transfer and pose-aware retrieval), which are challenging for entangled models, without any requirement for labels.

Acknowledgments We are grateful for support from NSERC (CGS-M-510941-2017) and Samsung Research.

References

- [1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2017. [2](#), [3](#), [6](#), [13](#), [14](#)
- [2] Abdul Fatir Ansari and Harold Soh. Hyperprior induced unsupervised disentanglement of latent representations. *arXiv preprint arXiv:1809.04497*, 2018. [2](#)
- [3] Matan Atzmon, Haggai Maron, and Yaron Lipman. Point convolutional neural networks by extension operators. *arXiv preprint arXiv:1803.10091*, 2018. [2](#)
- [4] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. [13](#)
- [5] Horace B Barlow et al. Possible principles underlying the transformation of sensory messages. *Sensory communication*, 1:217–234, 1961. [1](#)
- [6] Mikhail Belkin, Jian Sun, and Yusu Wang. Constructing laplace operator from point clouds in \mathbb{R}^d . In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 1031–1040. Society for Industrial and Applied Mathematics, 2009. [5](#), [20](#)
- [7] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013. [1](#)
- [8] Diane Bouchacourt, Ryota Tomioka, and Sebastian Nowozin. Multi-level variational autoencoder: Learning disentangled representations from grouped observations. *arXiv preprint arXiv:1705.08841*, 2017. [2](#)
- [9] Alexander M Bronstein, Michael M Bronstein, Leonidas J Guibas, and Maks Ovsjanikov. Shape google: Geometric words and expressions for invariant shape retrieval. *ACM Transactions on Graphics (TOG)*, 30(1):1, 2011. [6](#)
- [10] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in beta-vae. *arXiv preprint arXiv:1804.03599*, 2018. [2](#)
- [11] Tian Qi Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders. *arXiv preprint arXiv:1802.04942*, 2018. [2](#)
- [12] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180, 2016. [2](#)
- [13] Etienne Corman, Justin Solomon, Mirela Ben-Chen, Leonidas Guibas, and Maks Ovsjanikov. Functional characterization of intrinsic and extrinsic geometry. *ACM Transactions on Graphics (TOG)*, 36(2):14, 2017. [4](#)
- [14] Oren Dovrat, Itai Lang, and Shai Avidan. Learning to sample. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2760–2769, 2019. [3](#)
- [15] Babak Esmaeili, Hao Wu, Sarthak Jain, Alican Bozkurt, Narayanaswamy Siddharth, Brooks Paige, Dana H Brooks, Jennifer Dy, and Jan-Willem van de Meent. Structured dis-
entangled representations. *arXiv preprint arXiv:1804.02086*, 2018. [2](#), [4](#)
- [16] Sachin Sudhakar Farfade, Mohammad J Saberian, and Li-Jia Li. Multi-view face detection using deep convolutional neural networks. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 643–650. ACM, 2015. [3](#)
- [17] Lin Gao, Jie Yang, Yi-Ling Qiao, Yu-Kun Lai, Paul L Rosin, Weiwei Xu, and Shihong Xia. Automatic unpaired shape deformation transfer. In *SIGGRAPH Asia 2018 Technical Papers*, page 237. ACM, 2018. [2](#)
- [18] Shuyang Gao, Rob Brekelmans, Greg Ver Steeg, and Aram Galstyan. Auto-encoding total correlation explanation. *arXiv preprint arXiv:1802.05822*, 2018. [2](#)
- [19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. [1](#)
- [20] Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. Atlasnet: A papier-mache approach to learning 3d surface generation. In *CVPR 2018*, 2018. [3](#), [5](#)
- [21] 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. [12](#)
- [22] Naama Hadad, Lior Wolf, and Moni Shahar. A two-step disentanglement method. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 772–780, 2018. [2](#)
- [23] Sangchul Hahn and Heeyoul Choi. Disentangling latent factors with whitening. *arXiv preprint arXiv:1811.03444*, 2018. [2](#)
- [24] Pedro Hermosilla, Tobias Ritschel, Pere-Pau Vázquez, Àlvar Vinacua, and Timo Ropinski. Monte carlo convolution for learning on non-uniformly sampled point clouds. *arXiv preprint arXiv:1806.01759*, 2018. [2](#)
- [25] Irina Higgins, Loic Matthey, Xavier Glorot, Arka Pal, Benigno Uria, Charles Blundell, Shakir Mohamed, and Alexander Lerchner. Early visual concept learning with unsupervised deep learning. *arXiv preprint arXiv:1606.05579*, 2016. [1](#)
- [26] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β -vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. [2](#), [17](#)
- [27] IV Higgins and SM Stringer. The role of independent motion in object segmentation in the ventral visual stream: Learning to recognise the separate parts of the body. *Vision research*, 51(6):553–562, 2011. [1](#)
- [28] Haruo Hosoya. A simple probabilistic deep generative model for learning generalizable disentangled representations from grouped data. *arXiv preprint arXiv:1809.02383*, 2018. [2](#)

- [29] Du Q Huynh. Metrics for 3d rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, 35(2):155–164, 2009. 3
- [30] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 13
- [31] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Symposium on geometry processing*, volume 6, pages 156–164, 2003. 3
- [32] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. *arXiv preprint arXiv:1802.05983*, 2018. 2
- [33] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 13
- [34] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1
- [35] Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. *arXiv preprint arXiv:1711.00848*, 2017. 2, 4
- [36] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015. 6
- [37] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5
- [38] Chun-Liang Li, Manzil Zaheer, Yang Zhang, Barnabas Poczos, and Ruslan Salakhutdinov. Point cloud gan. *arXiv preprint arXiv:1810.05795*, 2018. 2, 3
- [39] Jiaxin Li, Ben M Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9397–9406, 2018. 3
- [40] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, Oct. 2015. 5, 12
- [41] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 13
- [42] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2391–2400. JMLR.org, 2017. 2
- [43] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and mathematics III*, pages 35–57. Springer, 2003. 5
- [44] Siddharth Narayanaswamy, Brooks Paige, Jan-Willem Van de Meent, Alban Desmaison, Noah Goodman, Pushmeet Kohli, Frank Wood, and Philip Torr. Learning disentangled representations with semi-supervised deep generative models. In *Advances in Neural Information Processing Systems*, pages 5925–5935, 2017. 4
- [45] Charlie Nash and Chris KI Williams. The shape variational autoencoder: A deep generative model of part-segmented 3d objects. In *Computer Graphics Forum*, volume 36, pages 1–12. Wiley Online Library, 2017. 2
- [46] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 13
- [47] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine learning in python. *CoRR*, abs/1201.0490, 2012. 6, 12, 13
- [48] David Pickup, Xianfang Sun, Paul L Rosin, Ralph R Martin, Z Cheng, Zhouhui Lian, Masaki Aono, A Ben Hamza, A Bronstein, M Bronstein, et al. Shape retrieval of non-rigid 3d human models. *International Journal of Computer Vision*, 120(2):169–193, 2016. 6
- [49] Gerard Pons-Moll, Javier Romero, Naureen Mahmood, and Michael J. Black. Dyna: A model of dynamic human shape in motion. *ACM Transactions on Graphics, (Proc. SIGGRAPH)*, 34(4):120:1–120:14, Aug. 2015. 5, 12
- [50] 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, 3, 13
- [51] 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
- [52] Siamak RavaniBakhsh, Jeff Schneider, and Barnabas Poczos. Deep learning with sets and point clouds. *arXiv preprint arXiv:1611.04500*, 2016. 2
- [53] Martin Reuter, Franz-Erich Wolter, and Niklas Peinecke. Laplace–beltrami spectra as shape-dna of surfaces and solids. *Computer-Aided Design*, 38(4):342–366, 2006. 5
- [54] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014. 1
- [55] Adrià Ruiz, Oriol Martinez, Xavier Binefa, and Jakob Verbeek. Learning Disentangled Representations with Reference-Based Variational Autoencoders. working paper or preprint, Oct. 2018. 2
- [56] Qingyang Tan, Lin Gao, Yu-Kun Lai, and Shihong Xia. Variational autoencoders for deforming 3d mesh models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5841–5850, 2018. 2
- [57] Jakub M Tomczak and Max Welling. Vae with a vampprior. *arXiv preprint arXiv:1705.07120*, 2017. 6
- [58] Bruno Vallet and Bruno Lévy. Spectral geometry processing with manifold harmonics. In *Computer Graphics Forum*, volume 27, pages 251–260. Wiley Online Library, 2008. 5
- [59] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. Learning localized generative models for 3d point clouds via graph

- convolution. In *International Conference on Learning Representations*, 2019. 2
- [60] Gl Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *CVPR*, 2017. 5, 12
- [61] Satoshi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of research and development*, 4(1):66–82, 1960. 2
- [62] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao. Spidercnn: Deep learning on point sets with parameterized convolutional filters. *arXiv preprint arXiv:1803.11527*, 2018. 2
- [63] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 3, 2018. 3
- [64] Shengjia Zhao, Jiaming Song, and Stefano Ermon. The information autoencoding family: A lagrangian perspective on latent variable generative models. *arXiv preprint arXiv:1806.06514*, 2018. 2
- [65] 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. 5, 12

Appendix

A. Dataset Details

A.1. MNIST Dataset

As a simple dataset on which to test our method, we generate point clouds from the greyscale MNIST images. We first produce triangle meshes by placing vertices at each pixel, with the x and y coordinates normalized by image width, and the z coordinate given by 0.1 times the normalized pixel value. Edges are added between each horizontal and vertical neighbour, as well as one diagonal, thus defining 2 triangles per set of four vertex neighbours. This is essentially the mesh of the height field defined by the image of the digit. We then threshold the mesh, deleting any vertices with a z height value less than 0.01. Finally, viewing the mesh as a graph, we take the largest connected component, giving us the final triangle mesh from which we sample our point clouds. This resulted in 59483 training and 9914 testing meshes with spectra. For the MNIST dataset, we fixed $N_T = 2000$, $N_\lambda = 40$, and $N_S = 1000$.

A.2. MPI Dyna Dataset

The Dyna dataset [49] consists of 3D scans of 10 individuals performing various sequences of simple actions (e.g., holding up their arms). In total, the dataset contains approximately 40K triangle meshes of multiple body types in a large variety of poses. During training, we fixed the total number of samples to $N_T = 6000$, the spectrum length as $N_\lambda = 100$, and the size of the input point clouds as $N_S = 2000$ for the Dyna dataset.

A.3. SMAL-derived Dataset

Using the SMAL model [65], we generated a dataset of animal meshes, including random body types and articulated motions. In detail, we use the fitted multivariate Gaussians computed by the authors of SMAL, which each act as a distribution over clusters of the shape parameters of the same animal species. We generate 3200 meshes for each of the five categories by sampling from each cluster distribution. Following the generation procedure in other works [21], we then sample the pose (here, the joint angles) via a Gaussian distribution with a standard deviation of 0.2. We then split the resulting dataset into 15000 training and 1000 testing meshes (each comprised of equal numbers of meshes per species). For SMAL, the total number of samples was $N_T = 8000$, the spectrum length was $N_\lambda = 50$, and the size of the input point clouds during training was $N_S = 1500$.

A.4. SMPL-derived Dataset

Similar to the dataset derived from SMAL, we generate a dataset of human meshes with random body types and articulations via the SMPL model [40]. We largely follow

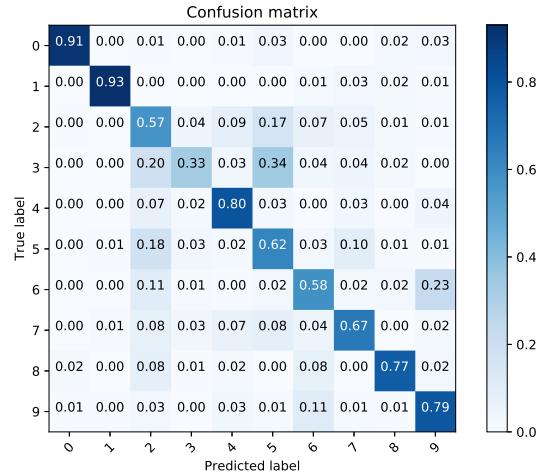


Figure 10. Confusion matrix of a shallow classifier, mapping mesh spectra to digit identity. Note the main confusions, e.g., between 9 and 6, match our intuition for the shape of the digit geometrically.

the protocol from 3D-CODED [21]. Briefly, we sampled 20500 meshes from each of the male and female models using random samples from the SURREAL dataset [60]. We augmented this data with 3100 meshes of ‘‘bent’’ humans per gender, using the alterations from Groueix et al [21]. We then assigned 500 unbent and 100 bent meshes (per gender) to the held-out test set, and the remaining meshes to the training set. This resulted in 45992 meshes and 1199 testing meshes after spectral calculations. Using these meshes, we derived point clouds with $N_T = 8000$, $N_\lambda = 40$, and $N_S = 2000$.

B. Spectral Geometry Intuition

In this section, we provide some intuition for the geometric meaning of the spectrum and why it can be used as a geometrically disentangled prior for shape representation, particularly for shapes that undergo isometric articulated pose transforms.

For MNIST, an obvious question is how the intrinsic geometry of a digit’s shape captured by its LBO spectrum is related to its semantic label (i.e., numeric value). Intuitively, a natural notion of the shapes of the digits should be closely related to their numeric identity, while minor perturbations that change the style of the digits should have less of an effect on the intrinsic shape. We examine this relation by training a simple classifier, which learns to map from the spectrum vector to the digit identity. We use a neural network with one hidden layer (size 100), using the ReLU non-linearity and otherwise default parameters from the scikit-learn library [47]. This obtains an accuracy of 0.69, suggesting that the intrinsic digit shape alone is capable of significant discriminatory power, but not as much as the complete shape information. The confusion matrix

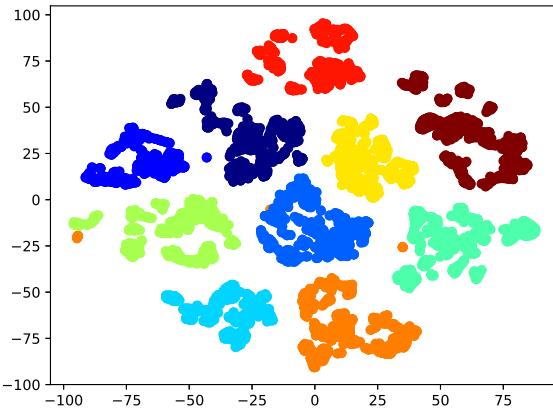


Figure 11. A t-SNE plot of the LBO spectra of the human shapes in the Dyna dataset. Each color corresponds to a different individual. Note the natural clusters formed by individuals.

is shown in Figure 10. The observed results agree with our intuition: for instance, 9 and 6 are close to rotations of each other, while 2 and 5 also clearly have very similar intrinsic shapes; unsurprisingly, these are relatively more common misclassifications. The most misclassified shapes, however, are 3 and 5, which differ only in the placement of the “bridge” between the lower curved part and the upper line making up the digits.

For datasets of some objects, such as people, articulations (i.e., non-rigid changes in pose for a single person) are nearly isometric transformations. Just as one naturally divides rigid and non-rigid deformations of objects, so too can one separate isometries from non-isometric deformations. To illustrate, in Figure 11, a t-Distributed Stochastic Neighbor Embedding (t-SNE) [41] plot (via scikit-learn [47]) shows the embeddings of the spectra of a portion of the Dyna dataset (every sixth shape in each activity sequence). Notice that the embedding naturally clusters by individual, as the intrinsic shape of each individual’s body is largely unchanged by the pose articulations of each action sequence. This corresponds to an intuitive view of intrinsic shape as being both continuous (as opposed to labelling each individual) and, in this case, articulation invariant.

C. Architectures

All models were implemented in PyTorch [46]. For exact values for each dataset, see Table 3.

C.1. Autoencoder Architecture

The encoder consists of a PointNet model [50], followed by fully-connected (FC) layers. The convolutional layers were of sizes $S_{AE,C}$. There were two affine transformers (“T-networks”) after the first and third layers, with convo-

lutional layers of size 64, 128, and 512, followed by an FC layer with hidden size 256. After the convolutional layers and affine transforms, 1D max pooling was applied, followed by processing by a FC network with hidden layers of sizes $S_{AE,FE}$. The final latent representation of the AE was of dimensionality L_{AE} . The decoder directly generated a fixed size point cloud (with $N_{AE,OUT}$ points), via a series of FC layers with hidden layer sizes $S_{AE,FD}$

We fixed the loss weighting parameters as follows: $\alpha_C = 0.75$, $\alpha_H = 0.5$. For MNIST and Dyna, FC and 1D convolutional layers utilized a BatchNorm layer [30] and the ReLU non-linearity between linear transforms. For SMPL and SMAL, no normalization and layer normalization [4] were used, respectively, in the decoder (similar to the encoder-only usage of batch norm in prior work [1]). Data augmentation consisted of random rotations about the “height” axis of the models (e.g., see Figure 15), with rotations in $[0, 2\pi]$ for MNIST and Dyna and in $[-\pi/6, \pi/6]$ for SMPL and SMAL. A batch-size of S_B and a learning rate of $\eta_{AE,LR}$ was used, with Adam [33] as the optimizer, for N_{EP} epochs.

C.2. GDVAE Architecture

The encoder consists of two parts: one for the rotation R and one for the shape X . Each part consists of two sub-parts: a set of shared layers, followed by a separate network for the variational parameters μ and Σ (i.e., $z_R \sim \mathcal{N}(\mu(g_{shared,r}(R)), \Sigma(g_{shared,r}(R)))$ and $(z_E, z_I) \sim \mathcal{N}(\mu(g_{shared,x}(X)), \Sigma(g_{shared,x}(X)))$). For the rotation mapping, the shared layers consisted of FC layers with hidden sizes 300, 200, while the mean and variance mappings are each given by a single affine transform. For the X mapping, the shared layers consisted of FC layers with hidden sizes 1000, 750, and 500, while the mean and variance are parameterized by a FC network with one hidden layer of size 250 (MNIST/Dyna) or 300 (SMPL/SMAL).

The decoder also consisted of two parts: the rotation decoder (FC layers with sizes 200 and 300) and the shape decoder (FC layers of sizes 500, 750, and 1000). The spectral predictor defined on the latent intrinsic z_I space was an FC network with layer sizes S_λ .

Models used a batch-size of $S_{V,B}$ for $N_{V,EP}$ epochs, using Adam with a learning rate of 0.0001. A small L_2 weight decay with coefficient 5×10^{-6} was used for MNIST and Dyna. Data was again augmented by random rotations about the height axis, but limited to an angular magnitude within $[-\pi/5, \pi/5]$ for MNIST and Dyna, and $[-\pi/12, \pi/12]$ for SMPL and SMAL.

Concerning hyper-parameters, unless otherwise specified, we used a spectral weight of $\zeta = 1000$, a relative quaternionic loss weight of $w_Q = 10$, hierarchically factorized VAE loss coefficients of $\beta_1 = \beta_2 = \beta_3 = 1.0$, and a reconstruction loss weight of $\eta = \dim(X)$ (on MNIST

		MNIST	Dyna	SMAL	SMPL
AE	$S_{AE,C}$	50, 100, 200, 300, 400	50, 100, 200, 300, 400	50, 100, 200, 400, 500	50, 100, 200, 400, 500
	$S_{AE,FE}$	300	300	800	500
	L_{AE}	250	250	400	300
	$S_{AE,FD}$	300, 500, 800	300, 500, 800	800, 800, 2000	500, 800, 2000
	r_C, r_H	10, 1	10, 1	20, 0.1	20, 0.1
	$N_{AE,OUT}$	1000	2000	1500	2000
	S_B, N_{EP}	32, 100	32, 200	30, 500	20, 100
VAE	$\eta_{AE,LR}$	0.00005	0.00005	0.00025	0.00025
	S_λ	500, 400	500, 400	300, 300	300, 400
	$S_{V,B}$	200	200	250	250
	$N_{V,EP}$	150	150	200	100
	$ z_E , z_I $	5, 5	10, 10	8, 5	12, 5
	β_4, γ_I, w_J	50, 1, 1	25, 5, 5	50, 100, 10	50, 10, 10

Table 3. Architectural parameters for the models across different datasets. See Sections C.1 and C.2 for details concerning the AE and VAE parameters, respectively.

	Beta-VAE	Jacobian	Cov	TC	All
β_4	1	1	1	100	100
γ_I	0	0	100	0	100
w_J	0	100	0	0	100

Table 4. Hyper-parameter values varied in the test of various loss weightings. (See Figure 13 for visualization of results).

and Dyna), $\eta = 200$ (on SMPL), and $\eta = 1$ (on SMAL). On SMPL, we used $\zeta = 500$, however. Recall that the weights on the disentanglement penalties were written β_4 for the inter-group TC, γ_I for the inter-group covariance, and w_J for the Jacobian.

D. MNIST Disentanglement

D.1. MNIST Latent Interpolations

Compared to that of articulating shapes (e.g., humans or other animal), the geometric disentanglement of the latent MNIST digit representation is less intuitive. Often, however, we found that moving in z_E would tend to deform the digit in “style”, while transversing z_I would more affect digit scale/thickness and identity. Some examples are shown in Figure 12. For instance, in the first inset, moving in z_E simply shifts around the lines of the ‘8’ without changing its digit identity (deforming it stylistically), while moving in z_I horizontally squishes the ‘8’ into a ‘1’.

D.2. MNIST Classification Results

We considered the information stored in the disentangled latent space segments by examining the performance of a classifier trained on various combinations of them, as representations of the digits. We show the confusion matrices for a linear SVM classifier on these sub-components of the latent spaces in Figure 14. Notice that utilizing z_R only has the poorest performance (and that adding it to z_{EI} has little

effect), and that the most prominent mistakes when using only z_I are very similar to those in Figure 10 (e.g., mixing up 2, 3, 4; and 6, 9).

E. Rotation Disentanglement

We examine the performance of the deterministic autoencoder, visualizing a number of reconstructions in Figure 15. Qualitatively, the points sampled over the reconstructions are largely uniformly spread out. However, thin or protruding areas tend to have lower densities of points, an issue identified by other works [1].

One question is how the presence of the rotation quaternion affects the representation. We visualize this by “derotating” the shapes (shown in the last row of each set of shapes in Figure 15), where the R portion of their representation is set to the same value. We can see that the derotated shapes tend to approximately fall into two or three groups with similar orientation. To confirm this, we also sample random shapes, rotate them, and then embed their X encodings via t-SNE in Figure 16. Qualitatively, we can see that rotating the point sets does not appear to lead to a single representation X ; instead, it forms a small number of latent groups.

We also tried adding a “rotational consistency” penalty to the loss function, which was computed by placing an L_2 loss between the X representation of pairs of shapes that differ only by a rigid rotation (i.e., $\|X - X_r\|_2^2$, where X and X_r are generated by rotations of the same point cloud). However, this seemed to lead to higher reconstruction error and did not entirely prevent the differing representations forming across rotations. Furthermore, as noted in the main text, some shapes can naturally become a “new” shape, semantically speaking, due to rotation (e.g., a 9 becoming a

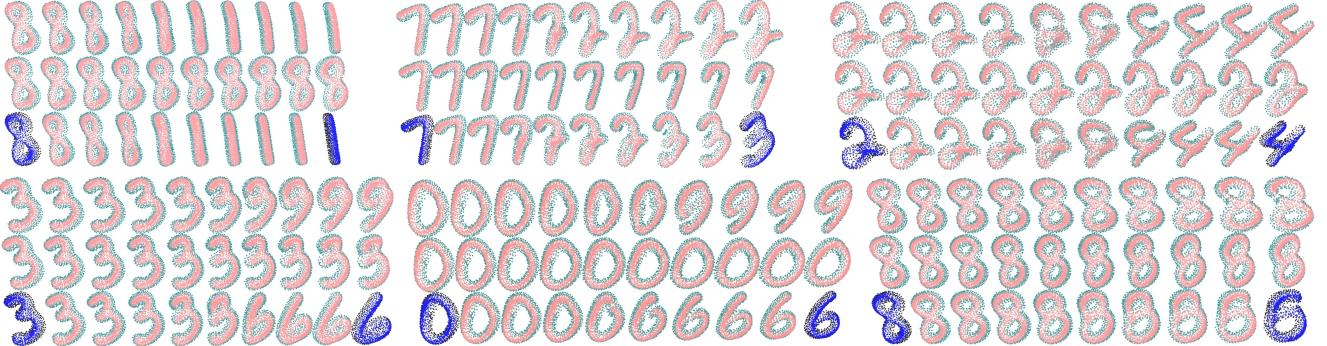


Figure 12. Disentangled latent interpolations for the MNIST dataset. Colours per digit denote depth. For each inset, we are interpolating between the blue-black digits in the bottom row. The first row corresponds to only moving through z_I (with z_E constant), while the second corresponds to only moving through z_E (holding z_I fixed). The third row traverses the full latent z space. In all cases, z_R is set to zero.

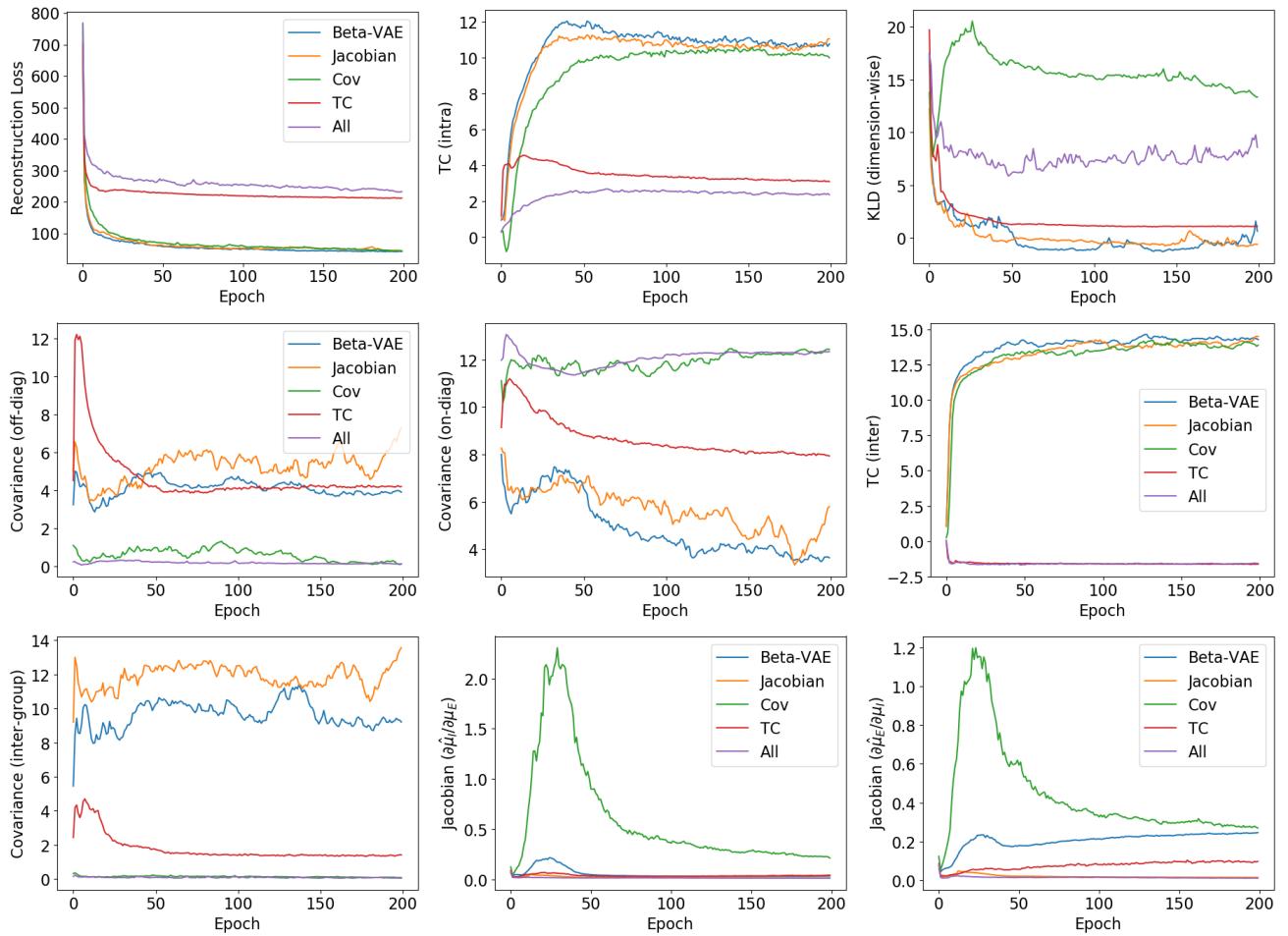


Figure 13. Empirical curves of loss terms during training across weight hyper-parameters on model learning. See Table 4 for weight values. Each colored curve represents a different set of weight values (i.e., model hyper-parameters). Top row: log-likelihood reconstruction loss, intra-group TC, dimension-wise KL divergence. Middle row: off-diagonal intra-group covariance, on-diagonal covariance terms (i.e., variances), inter-group TC. Bottom row: inter-group covariance, Jacobian penalty (intrinsics with respect to extrinsics), Jacobian penalty (extrinsics with respect to intrinsics).

6), which this penalty does not allow.

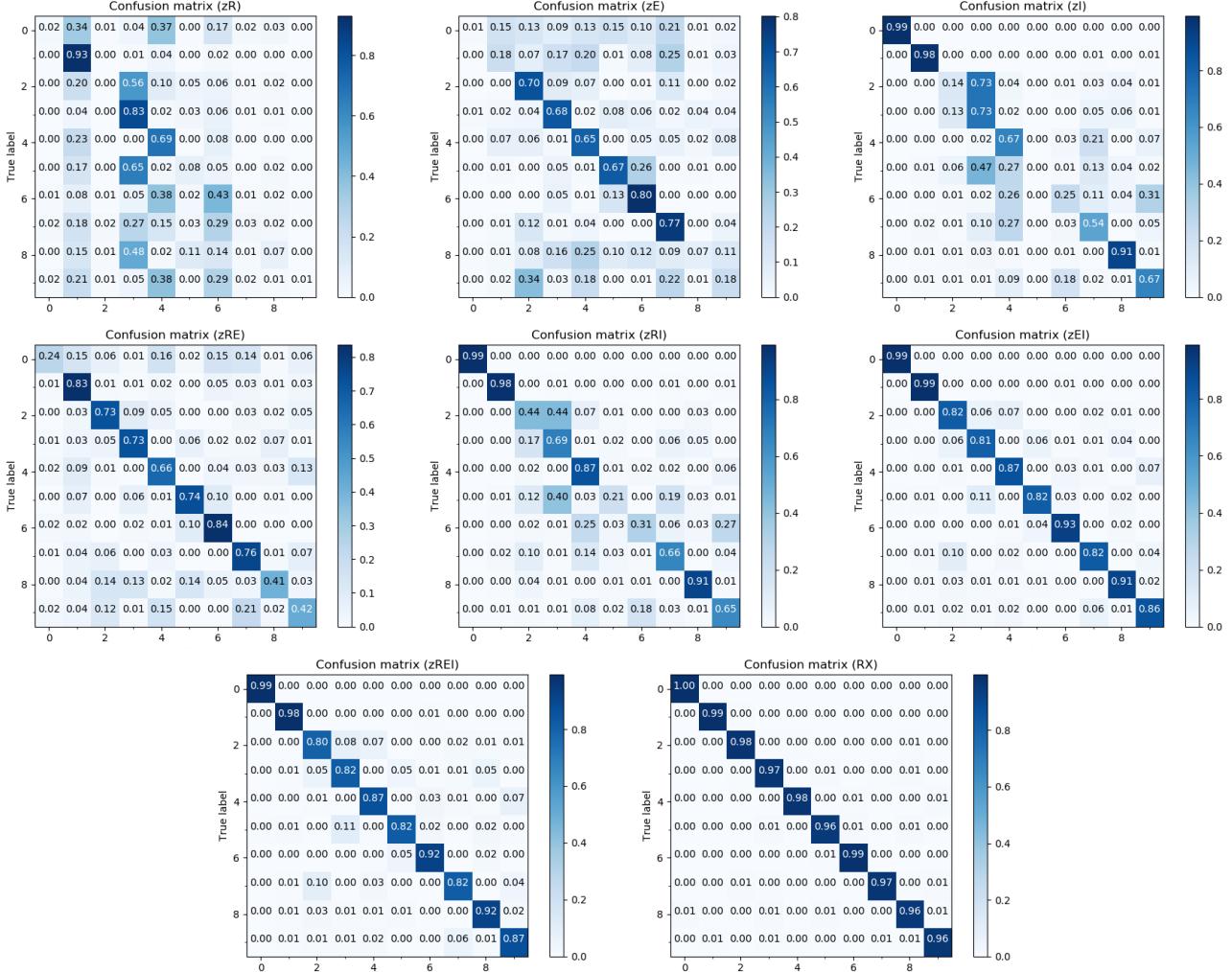


Figure 14. Confusion matrices for a linear SVM classifier accuracy on the MNIST test set. Top row: testing with z_R , z_E , z_I . Middle row: testing with z_{RE} , z_{RI} , z_{EI} . Bottom row: testing with, z_{REI} , (R, X) .

F. Model Analysis

F.1. Hyper-parameter Variation

Our generative model must balance three main terms: (1) autoencoding reconstruction, (2) latent prior sampling, and (3) disentanglement. We therefore trained five models on MNIST under five different hyper-parameter conditions (see Table 4) to showcase the relative effect of these loss weights. The choice of models was designed to check how the different penalties affected the metrics of disentanglement based on each loss (e.g., how penalizing covariance affects the pairwise Jacobian penalty), as well as autoencoding and generation.

Based on the results (loss curves are shown in Figure 13), we can make several observations. Firstly, a high TC penalty results in decreased inter-group covariance and Jacobian losses. However, while the covariance and Jacobian

penalties effectively reduce their own penalties, they struggle to reduce the TC or each other. Nevertheless, the TC penalty alone does not drive the covariance or Jacobian values as low as having a penalty on them directly does. This suggests that the TC may be a more powerful penalty, but it can still be complemented by the other approaches. One explanation for this effect is that the Jacobian penalty is fundamentally local (penalizing the expected change with respect to an infinitesimal perturbation around each data point separately) and the covariance penalty only reduces linear correlations, whereas the TC penalty is information-theoretic (i.e., able to detect non-linear relations) and considers the estimated latent probability distributions on a more global level.

On the other hand, the models with high TC show the worst log-likelihood for reconstructions. The model with high weights for all terms also has poor dimension-wise KL



Figure 15. Examples of autoencoder reconstructions across different rotations of the same object. Note that the encoding of the datum from the AE is not passed through the VAE. Colour denotes depth (z value). Within each group of shapes, the columns show different rotations of the left-most shape, while the top row shows input shapes (i.e., P , from the original data), the middle row shows the reconstruction \hat{P} , and the bottom row shows the derotated shape with the rotation component R set to the value given by the shape in the first column. Note that these models were trained with data augmentations across all rotations about the gravity axis.

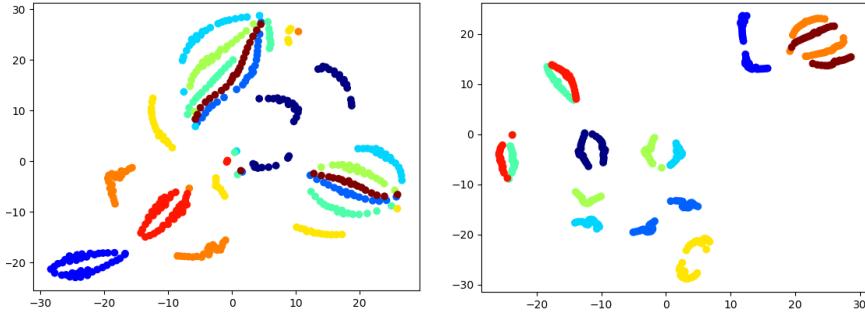


Figure 16. A t-SNE visualization of the rotated shapes from the deterministic AE. Left: plot of MNIST. Right: plot of Dyna. In both plots, 10 random shapes are selected, and rotated 40 times, evenly spread over $[0, 2\pi]$, with respect to the height axis. Each shape is then encoded by the AE, and only the X component of its representation is plotted. A single color corresponds to a single shape.

divergence, meaning the ability to generate novel samples may be compromised, though it is not as high as that of the covariance-penalized model. Though our observations are limited to this dataset, hyper-parameters, and architecture, they suggest that the disentanglement term can be in conflict with reconstruction and sampling (just as the latter two are

known to be in conflict with each other; e.g., [26]).

F.2. Disentanglement Penalty Ablation

To further examine the effect of the disentanglement penalties, in particular the covariance and Jacobian terms, we considered two ablation experiments on the SMPL and

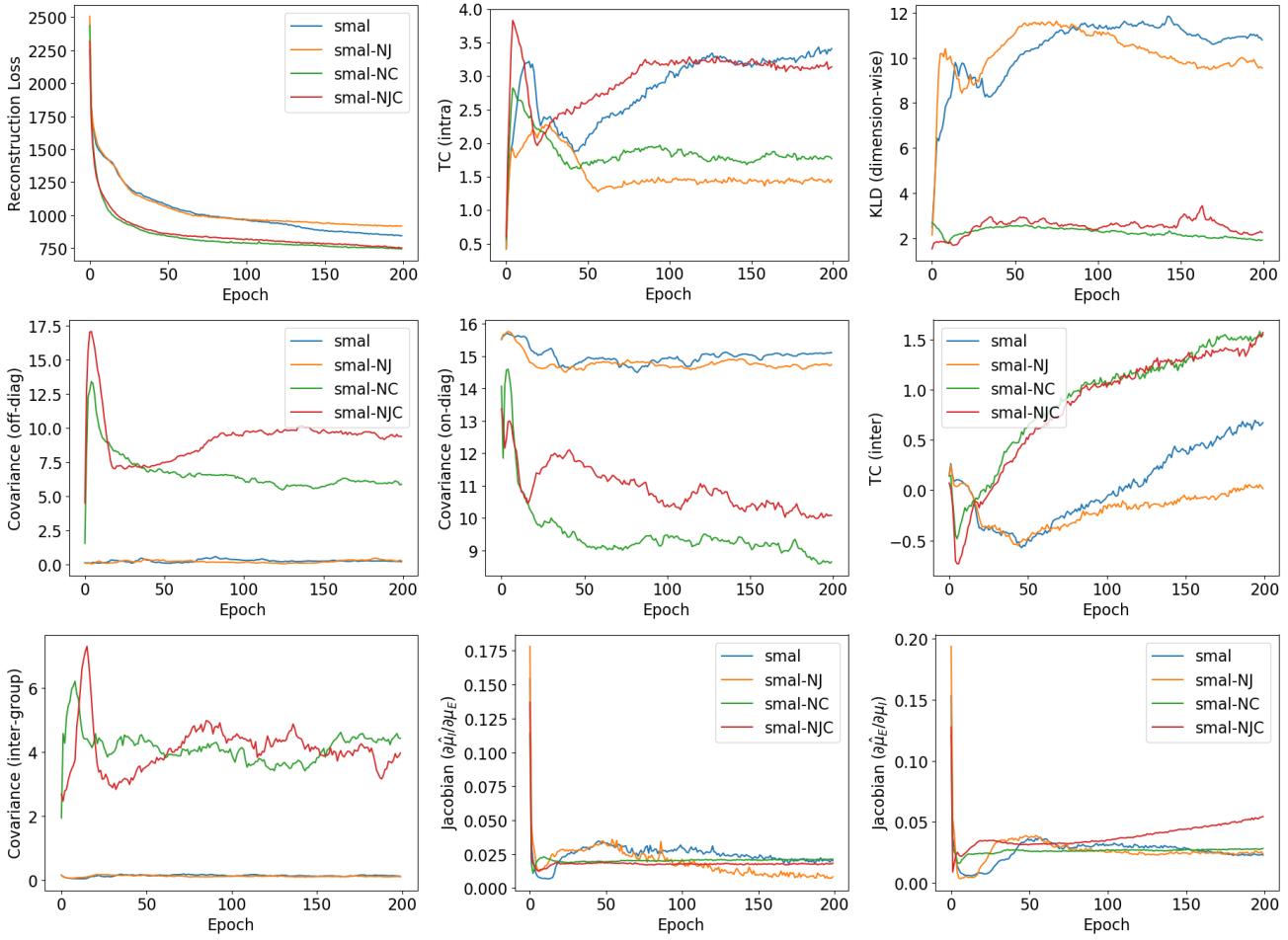


Figure 17. Empirical curves of loss terms for disentanglement ablations on SMAL. NC, NJ, and NJC mean no covariance, no Jacobian, or neither penalties, respectively. Each colored curve represents a different set of weight values (i.e., model hyper-parameters). Top row: log-likelihood reconstruction loss, intra-group TC, dimension-wise KL divergence. Middle row: off-diagonal intra-group covariance, on-diagonal covariance terms (i.e., variances), inter-group TC. Bottom row: inter-group covariance, Jacobian penalty (intrinsics with respect to extrinsics), Jacobian penalty (extrinsics with respect to intrinsics).

SMAL datasets: (1) looking at the loss curves of the estimated entanglement measures during training and (2) considering the pose-aware retrieval performance. We examined three conditions, in addition to the regular hyper-parameter values (REG): no Jacobian penalty (NJ), no covariance penalty (NC), and no Jacobian or covariance penalties (NJC).

The various loss terms are shown over training epochs in Figures 17 and 18, for SMAL and SMPL respectively. On SMAL, we see that REG and NJ have worse KL-divergence and reconstruction loss, but better inter-group TC and covariance. NJC scores the worst on all four entanglement measures, except $\partial\hat{\mu}_I/\partial\mu_E$ (where surprisingly NJ holds the smallest value). On SMPL, we see that NJ consistently has the highest Jacobian penalties, and that REG and NJ have the worst TC. On both datasets, we see that the covariance penalty is necessary to ensure the inter-group covari-

ance is small, since NC and NJC always have much higher inter-group covariance than NJ and REG.

We next considered the pose-aware retrieval task under the various ablation conditions. Results are shown in Table 5. Recall that low E_θ using z_E and low E_β using z_I are good (indicating z_I and z_E hold intrinsic shape and pose respectively), while low E_θ using z_I and low E_β using z_E are not (as it means entanglement is present). For SMAL, we see that REG has the best E_β using z_I , while NJC is worse than both NJ and NC. We can also see poor entanglement in the low E_β using z_E of NJC. The scores for E_θ are poor across all the representations, even including X . NC has the best E_θ using z_E , but it has a worse (i.e., lower) E_β using z_E , suggesting increased entanglement. For SMPL, REG has the best E_θ using z_E , but NC and NJ have better E_β values with z_I . However, NC and NJ also have lower E_θ with z_I , meaning pose is entangled with intrinsic shape.

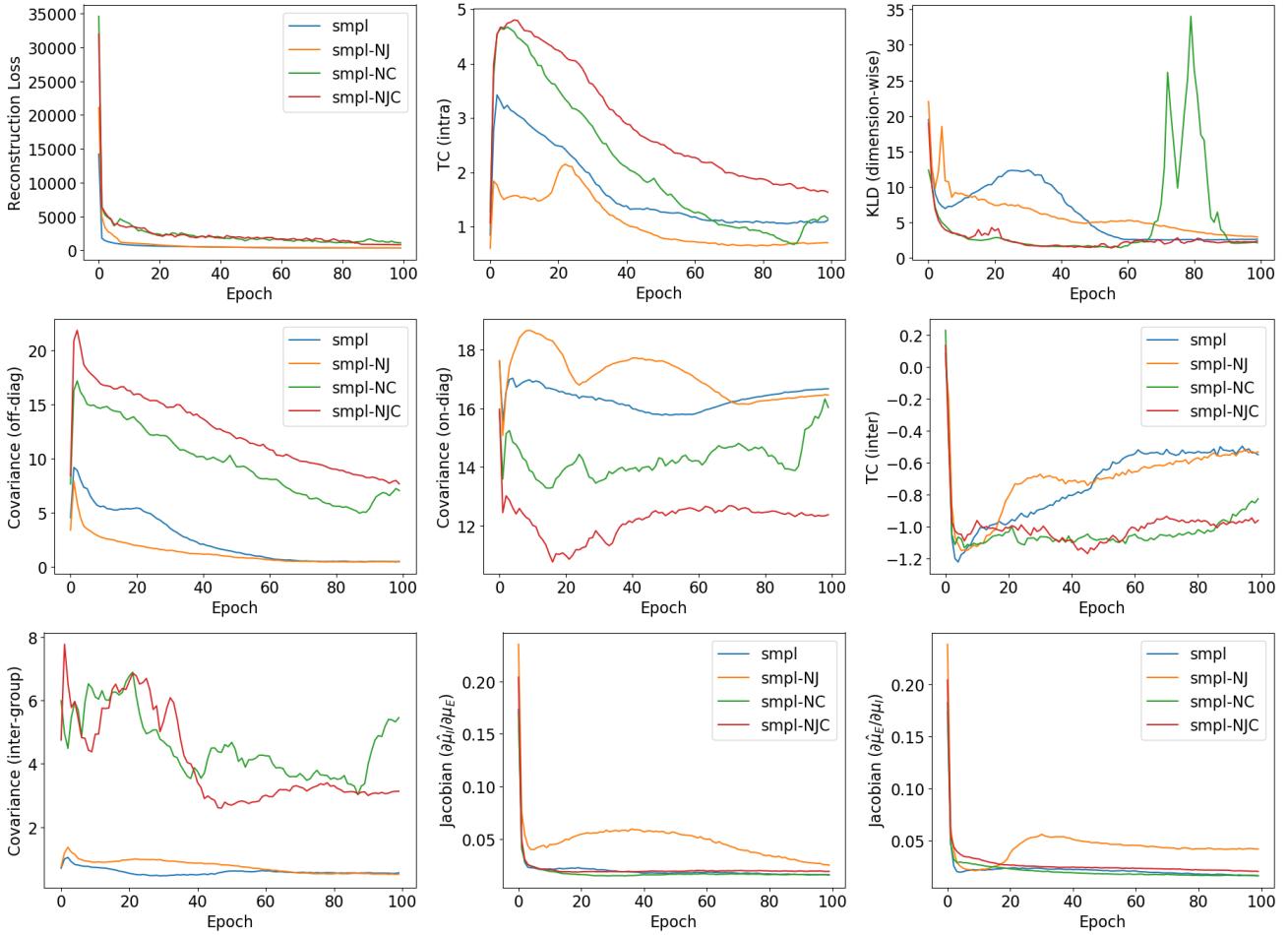


Figure 18. Empirical curves of loss terms for disentanglement ablations on SMPL, NC, NJ, and NJC mean no covariance, no Jacobian, or neither penalties, respectively. Top row: log-likelihood reconstruction loss, intra-group TC, dimension-wise KL divergence. Middle row: off-diagonal intra-group covariance, on-diagonal covariance terms (i.e., variances), inter-group TC. Bottom row: inter-group covariance, Jacobian penalty (intrinsics with respect to extrinsics), Jacobian penalty (extrinsics with respect to intrinsics).

It's also worth noting that NJC on both SMAL and SMPL has lower error on both E_β and E_θ when using z . This suggests that the increased disentanglement penalties make holding information harder in general (i.e., allowing more entanglement can increase reconstruction and thus retrieval performance).

G. Pose-Aware Retrieval Variance

In Table 6, we consider the standard error of the mean (SEM) for the retrieval experiments. Each model was trained three times with the same hyper-parameters (to account for training stochasticity), and then each model was run three times to account for randomness in the point set sampling of the input shapes. We show the SEM across models (i.e., trainings) after averaging over point samplings per model. Since each model has its own SEM over samplings, we display the maximum SEM across models. No-

tice that all SEMs are less than 0.01, except for two (over model trainings): the E_β when using z and z_I on SMAL, which are the most unstable retrieval performances. In general, training instability is a useful consideration for future work.

H. Pose-Aware Retrieval with Spectral Noise

We also investigated the effect of spectral noise on model performance. We considered three forms of noise: (1) by directly injecting Gaussian multiplicative noise into the spectra, (2) replacing the spectrum with Gaussian noise (mean zero, sigma 50), and (3) by extracting the LBO from sampled point clouds (rather than meshes). To add the noise in (1), we multiply each eigenvalue with independent noise $\xi \sim \mathcal{N}(1, \sigma^2)$, and then enforced non-negativity (via clipping) and monotonicity of the spectrum (by re-sorting the spectrum). To obtain the spectra in (3), we used a simple

		X	z	z_E	z_I
SMAL	E_β	0.641	0.743	0.975	0.645
	E_θ	0.938	0.983	0.983	0.993
SMAL-NJ	E_β	0.642	0.829	0.980	0.734
	E_θ	0.938	0.979	0.980	0.996
SMAL-NC	E_β	0.642	0.670	0.962	0.656
	E_θ	0.938	0.966	0.969	0.991
SMAL-NJC	E_β	0.642	0.661	0.834	0.891
	E_θ	0.938	0.967	0.978	0.982
SMPL	E_β	0.856	0.922	0.997	0.928
	E_θ	0.577	0.726	0.709	0.947
SMPL-NJ	E_β	0.858	0.907	1.006	0.895
	E_θ	0.578	0.695	0.812	0.908
SMPL-NC	E_β	0.855	0.908	0.995	0.905
	E_θ	0.578	0.727	0.836	0.909
SMPL-NJC	E_β	0.857	0.888	0.992	0.921
	E_θ	0.578	0.693	0.722	0.965

Table 5. Different retrieval scores under various disentanglement penalty ablation conditions. NJ, NC, and NJC mean no Jacobian, no covariance, and neither Jacobian nor covariance cases respectively. Note that differences between scores under X are due to the random samplings of points from the shape; hence, each score is obtained by running the process three times (across point samplings). However, only the scores for SMAL and SMPL are averaged over multiple training runs.

		X	z	z_E	z_I
SMAL	E_β	0.0006	0.0282	0.0035	0.0381
	E_θ	0.0003	0.0015	0.0015	0.0015
(M-SEM)	E_β	0.0003	0.0048	0.0015	0.0032
	E_θ	0.0003	0.0032	0.0058	0.0059
SMPL	E_β	0.0028	0.0020	0.0030	0.0024
	E_θ	0.0008	0.0005	0.0006	0.0008
(S-SEM)	E_β	0.0015	0.0013	0.0020	0.0053
	E_θ	0.0008	0.0017	0.0022	0.0070

Table 6. SEMs across model training runs (M-SEM) and shape samplings (S-SEM). Model training SEMs are computed over the mean of the shape sampling runs; shape sampling SEMs are computed by taking the SEM over point samplings per model, and then the maximum SEM across models.

Laplacian computed from an affinity matrix, via a radial basis function kernel on the inter-point L_2 distance, with bandwidth chosen as $b_\sigma = d_N^{-1/(2+\xi)} / 4$, where d_N is the mean distance of each point to its nearest neighbour and $\xi = 0.01$ (similar to [6]).

We note that the shapes (input point clouds) themselves do not have any additional noise (compared to the standard experimental setup), only the spectra do. Thus, poorer performance may manifest itself as increased retrieval accuracy with the *wrong* latent space segment (e.g., lower E_β when retrieving with z_E).

Results are shown in Table 7. However, we find that the

		X	z	z_E	z_I
SMAL	E_β	0.641	0.724	0.961	0.666
	E_θ	0.937	0.969	0.980	0.981
$\sigma = 0.05$	E_β	0.641	0.723	0.900	0.868
	E_θ	0.938	0.969	0.979	0.994
$\sigma = 0.1$	E_β	0.636	0.810	0.893	0.838
	E_θ	0.938	0.979	0.988	0.996
$\sigma = 0.2$	E_β	0.643	0.822	0.791	0.913
	E_θ	0.938	0.977	0.985	0.998
SMAL-R	E_β	0.639	0.629	0.783	0.929
	E_θ	0.939	0.973	0.973	0.995
SMAL-P	E_β	0.641	0.698	0.636	0.895
	E_θ	0.938	0.980	0.984	0.993
$n_p = 1.2K$	E_β	0.857	0.910	1.000	0.913
	E_θ	0.578	0.735	0.774	0.966
$\sigma = 0.05$	E_β	0.856	0.909	0.979	0.929
	E_θ	0.578	0.694	0.720	0.979
$\sigma = 0.1$	E_β	0.858	0.919	0.980	0.925
	E_θ	0.578	0.710	0.810	0.946
$\sigma = 0.2$	E_β	0.857	0.933	0.924	0.987
	E_θ	0.579	0.694	0.826	0.943
SMPL-R	E_β	0.856	0.946	0.972	0.948
	E_θ	0.577	0.673	0.731	0.954
SMPL-P	E_β	0.856	0.926	0.990	0.944
	E_θ	0.578	0.695	0.739	0.982

Table 7. Retrieval results in the presence of spectral noise. Each row corresponds to retrieval results in the presence of spectral noise either due to multiplicative Gaussian noise (with strength σ), replacing the spectrum with independent Gaussian noise (mean zero, sigma 50; denoted “-R”), or using an LBO estimated from a point cloud (denoted “-P”, using a point cloud of size n_p , either 1200 or 2000).

network can tolerate moderate spectral noise, with decreasing performance as noise increases. For instance, one can see E_β with z_I on SMAL and E_θ with z_E on SMPL degrade as σ increases. Very extreme noise, as when replacing the spectrum with Gaussian random values (“SMAL-R” and “SMPL-R” in Table 7), destroys the disentanglement, as the network no longer has access to the isolated intrinsics. This is similar to ablating the spectral loss, except that predicting the random spectrum adds additional burden on z_I . Finally, for the spectra extracted from the point cloud Laplacians, the network degrades slightly on SMPL, compared to using the mesh LBO, but much more so on SMAL.

In practice, we note that our method does not require spectra at test time (for inference or novel sample generation). However, it will be affected by noise in the training data (whether in the meshes, spectra, or point clouds). One approach to improve generalization to noisy point clouds at test time is to use additional noise for data augmentation while training.