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) [18] or variational autoencoders (VAEs) [48, 30]. These methods are not only capable of generating novel examples of data points, but also produce a latent space that 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 [6] is a major tenet of representation learning, that closely aligns with human reasoning, and is supported by neuroscientific findings [4, 25, 23]. 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 $\beta$-VAE [24, 9], 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 [55]. Minimizing TC corresponds to minimizing the information shared among variables, making it a powerful disentanglement technique [17, 10, 29]. 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 [14].

Other examples of disentanglement algorithms include information-theoretic methods in GANs [11], latent whitening [21], covariance penalization [31], and Bayesian hyperpriors [2]. A number of techniques also utilize known groupings or discrete labels of the data [26, 7, 49, 20]. 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 outside 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 [46]. PointNet [44] 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., [45, 22, 56, 3]), but this remains an open problem.

Several generative models of point clouds exist: Nash and Williams [40] utilize a VAE on data of 3D part segmentations and associated normals, whereas Achlioptas et al. [1] use a GAN. Li et al. [34] adopt a hierarchical sampling approach with a more general GAN loss, while Valsesia et al. [53] 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., [37, 58]).

Two recent related works utilize meshes for deformation-aware 3D generative modelling. Tan et al. [50] utilize latent manipulation to perform a variety of tasks, but does not explicitly separate pose and shape. Gao et al. [16] 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 $(\hat{R}, \hat{X})$ 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. $(\hat{R}, \hat{X})$ 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 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.

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, $(\hat{R}, \hat{X})$ 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., [28, 15]), 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, 35, 13], we utilize a form of the Chamfer distance as our main measure of similarity. We define the max-average function

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

where $\alpha$ 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 [57]. We then use the point cloud loss

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

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

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

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 $SO(3)$ [27]; 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 [27] $\mathcal{L}_Q = 1 - |q \cdot \tilde{q}|$ 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 [44], 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 [34, 19]). 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 $\lambda$, 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 (HF-VAE) defined by Esmaeili et al. [14], but the third term also includes a covariance penalty motivated by the Disentangled Inferred Prior VAE (DIP-VAE) [31] 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., $z = (z_1, \ldots, z_{|G|})$, 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. [14] showed that the prior-matching term of the VAE objective (i.e., $\mathcal{D}_{KL} [q_\phi(z|x) \mid\mid p(z)]$) can be hierarchically decomposed as

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

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

As far as disentanglement is concerned, the main term enforcing inter-group independence (via the TC) is the one weighted by $\beta_4$. 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 [39].

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. [31], who discuss how disentanglement can be better controlled by introducing a penalty 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 $\hat{C}$ 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 $\mu_g$ as the part of $\mu_\phi(x)$ corresponding to group $g$ (i.e., parameterizing the approximate posterior over $z_g$) and define

$$
\mathcal{L}_{COV} = \gamma I \sum_{g \neq \tilde{g}} \sum_{i,j} \left| \hat{C}(\mu_g, \mu_{\tilde{g}})_{ij} \right| \tag{5}
$$
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 [31], 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

$$
\mathcal{L}_J = \max_{g \neq \tilde{g}} \left\| \frac{\partial \hat{\mu}_{\tilde{g}}}{\partial \mu_g} \right\|_F^2, \tag{6}
$$

where $\hat{X}$ is the decoded shape, $\hat{\mu}_g$ represents group $g$ from $\mu_\phi(\hat{X})$, 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 [12]. 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 $\lambda$ can be formally described by viewing a shape as a 2D Riemannian manifold $(M, g)$ embedded in 3D, with point clouds being viewed as random samplings from this surface.

Given the spectrum $\lambda$ of a shape, we wish to compute a loss with respect to a predicted spectrum $\hat{\lambda}$, treating each as a vector with $N_\lambda$ elements. The LBO spectrum has a very specific structure, with $\lambda_i \geq 0 \forall i$ and $\lambda_i \geq \lambda_k \forall j > k$. Analogous to frequency-space signal processing, larger elements of $\lambda$ 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 [52]. Due to this structure, a naive vector space loss function on $\lambda$ (e.g., $L_2$) will over-weight learning the higher frequency elements of the spectrum. We suggest that the lower portions of $\lambda$ 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:

$$L_S(\lambda, \hat{\lambda}) = \frac{1}{N_\lambda} \sum_{i=1}^{N_\lambda} \frac{|\lambda_i - \hat{\lambda}_i|}{i},$$

(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., [47]), which asserts that spectrum elements increase approximately linearly, for large enough $i$.

### 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:

$$L_V = \frac{1}{D}||X - \hat{X}||_2^2 + w_Q L_Q,$$

(8)

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

$$\mathcal{L} = \eta L_V + L_{HF} + L_{COV} + w_J L_J + \zeta L_S.$$  

(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 [33], the MPI Dyna dataset of human shapes [43], a dataset of animal shapes from the Skinned Multi-Animal Linear model (SMAL) [59], and a dataset of human shapes from the Skinned Multi-Person Linear model (SMPL) [36] via the SURREAL dataset [54]. 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. [19]. 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 [38], as it provides a more reliable result than algorithms utilizing point clouds (e.g., [5]). We thus obtain a spectrum $\lambda$ as a $N_\lambda$-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.

\[
\begin{array}{cccccccc}
  z_R & z_E & z_I & z_{RE} & z_{RI} & z_{EI} & z & S
  \\
  0.32 & 0.47 & 0.60 & 0.64 & 0.68 & 0.88 & 0.88 & 0.98
\end{array}
\]

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 \( S = (R, X) \).

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 extrinsics \( z_I \). 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., [8, 42]), 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 ($\beta$) and articulated pose parameters (Rodrigues vectors at joints, $\theta$), 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 $\beta$ or $\theta$ 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 $\beta$ and $\theta$ values of the query $P_Q$ ($\beta_Q$, $\theta_Q$) are to those of a retrieved shape $P_R$ ($\beta_R$, $\theta_R$). 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 $\theta$ to the equivalent unit quaternion $q(\theta)$, and then compute $E_\theta(P_Q, P_R) = L_Q(q(\theta_Q), q(\theta_R))$. 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 $E_\beta$ and a low $E_\theta$, while using $z_I$ should have a high $E_\theta$ and a low $E_\beta$.

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 $E_\beta$ for $z_I$ is much lower than for $z_E$ (and $z$ on SMAL), while the $E_\theta$ 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 $E_\beta$ of $z_E$ is much higher than that of $z$, as is the $E_\theta$ of $z_I$. We emphasize that $E_\beta$ and $E_\theta$ 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:** $z_I \sim N(0, I)$, fixed $z_E$; **lower row:** $z_E \sim N(0, I)$, fixed $z_I$. Colors denote depth (distance from the camera).

<table>
<thead>
<tr>
<th></th>
<th>$X$</th>
<th>$z$</th>
<th>$z_E$</th>
<th>$z_I$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMAL</td>
<td>$E_\beta$</td>
<td>0.641</td>
<td>0.743</td>
<td>0.975</td>
</tr>
<tr>
<td></td>
<td>$E_\theta$</td>
<td>0.938</td>
<td>0.983</td>
<td>0.983</td>
</tr>
<tr>
<td>SMPL</td>
<td>$E_\beta$</td>
<td>0.856</td>
<td>0.922</td>
<td>0.997</td>
</tr>
<tr>
<td></td>
<td>$E_\theta$</td>
<td>0.577</td>
<td>0.726</td>
<td>0.709</td>
</tr>
</tbody>
</table>

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).

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


