Manifold Learning in Quotient Spaces

Eloi Mehr, André Lieutier, Fernando Sanchez Bermudez, Vincent Guitteny, Nicolas Thome, Matthieu Cord

Manifold Learning in Quotient Spaces

Éloi Mehr[1], André Lieutier , Fernando Sanchez Bermudez , Vincent Guitteny , Nicolas Thome[1], and Matthieu Cord[1]

[1]LIP6, UPMC Sorbonne Universités

Abstract

When learning 3D shapes we are usually interested in their intrinsic geometry rather than in their orientation. To deal with the orientation variations the usual trick consists in augmenting the data to exhibit all possible variability, and thus let the model learn both the geometry as well as the rotations. In this paper we introduce a new autoencoder model for encoding and synthesis of 3D shapes. To get rid of undesirable input variability our model learns a manifold in a quotient space of the input space. Typically, we propose to quotient the space of 3D models by the action of rotations. Thus, our quotient autoencoder allows to directly learn in the space of interest, ignoring side information. This is reflected in better performances on reconstruction and interpolation tasks, as our experiments show that our model outperforms a vanilla autoencoder on the well-known Shapenet dataset. Moreover, our model learns a rotation-invariant representation, leading to interesting results in shapes co-alignment. Finally, we extend our quotient autoencoder to quotient by non-rigid transformations.

1. Introduction

Manifold learning and generative models are major research topics to learn from a dataset a latent space representing the input space [4, 21]. They are useful tools, not only to grasp the hidden structure of the data, but also to generate new plausible data, either by sampling or by non-linear interpolation, or to complete missing information by inferring unobserved variables.

In this work we focus on manifold learning of 3D shapes. Such data is often represented in a high-dimensional real vector space X. Manifold learning aims at representing the data by learning an embedding in a low-dimensional space. However, explicit representations of 3D shapes (point clouds, voxels, meshes, CAD[1] models) are given through coordinates and therefore in a given pose. These natural representations carry two pieces of information: the "intrinsic geometry" of the shape, *i.e*. its geometry up to rigid transformations, together with the pose information itself. While mathematically clear, intrinsic geometries are the quotient of X by the action of the group of rigid transformations, following Kendall's shape spaces [18], the quest for simple and parsimonious explicit representations of them remains elusive. To "disentangle" both pieces of information, and learn an explicit representation of intrinsic geometry, one possibility consists in seeing the shape as a metric space, either through heat kernel (or Laplacian) based signatures [23, 29] or, in the case of a point cloud, by giving Euclidean distances between each pair of points. While interesting for some applications, these representations are not parsimonious. One could also suggest rotation-invariant spherical harmonics [17], but getting rid of pose information in them makes us lose most geometric information.

In this paper we propose a strategy to bridge the gap between pre-processing of 3D datasets using co-analysis tech-

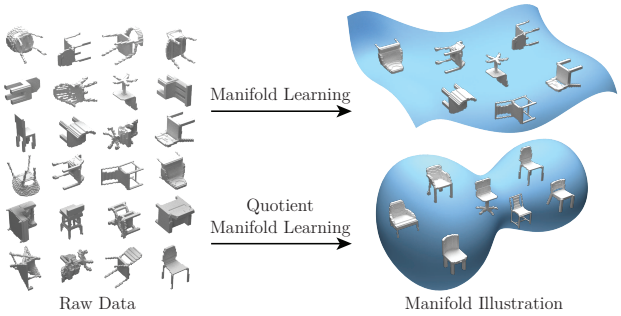

Figure 1: Misaligned dataset is learned in the space of 3D shapes quotiented by rotations, thus the quotient autoencoder focuses on the intrinsic geometry only. As a result it reconstructs better than a vanilla autoencoder, and is also able to co-align the shapes.

[1]Computer assisted design.

niques and 3D manifold learning. We introduce a quotient model that allows to directly learn the manifold in the space of the geometry, independently of the pose of the input 3D shape in the dataset (see Figure 1). To produce this parsimonious intrinsic geometry representation, we design a variant of the autoencoder [14, 33], called **quotient autoencoder** (abbreviated **QAE**), based on a dedicated loss function working in the quotient space $X/\mathcal{G}$, where $\mathcal{G}$ is a group, typically the group of rotations. Our method, by some economy principle, allows a shared pose to emerge by implicitly constraining the decoder to select a pose at the reconstruction stage. To further support this behavior, we introduce a specific orbit pooling in our encoder. We also derive a training scheme adapted for both discrete and continuous transformation groups.

We provide empirical results to demonstrate that our QAE is more efficient than a vanilla autoencoder trained with data augmentation. We also provide several experiments to highlight the ability of our QAE to automatically co-align shapes, and we illustrate realistic interpolations of misaligned shapes. Additionally, we show that our framework can be extended to non-rigid transformations with other applications, such as feature matching.

2. Background

Autoencoders. Given a dataset representing samples from a same class, *e.g.* a dataset of cars, autoencoders [14, 33] aim at learning a mapping, modeled as a feedforward deep neural network, between the original input space and a low-dimensional space. Moreover, autoencoders also learn a reverse mapping from the latent space to the original input space. They can notably be used to perform realistic non-linear interpolation, extract meaningful features in the data, compress the data into a compact representation, complete a partial shape, or sample new shapes (see [3] and also [20, 27] for extensions to a variational framework).

Co-alignment of shapes. Co-alignment consists in aligning a dataset of 3D models in a common frame. Many drawbacks are inherent to the existing methods. Some require manual supervision such as [34]. Others like [2, 7] are based on symmetries and correlations analysis, or try to minimize a continuous energy as reviewed in [30]. These methods are not adapted to shapes exhibiting high geometrical or topological variability. Some methods also need very clean meshes to be able to compute accurate features, as in [2]. Moreover, most of the approaches compute the alignment by a pairwise matching of each sample in the dataset with respect to a reference mesh, but are not able to leverage the whole dataset at once to compute a consistent frame.

3D generative learning. Several recent works in unsupervised learning have been dedicated to 3D shapes. [36] presents a convolutional deep belief network, which is used for shape completion and recognition tasks. Autoencoers have also been used, for instance in [12], where additionally to a 3D autoencoder a CNN is learned to predict the latent vector from an image. [35] extends this idea by replacing the autoencoder with a generative adversarial network (see [13]), and adds a variational regularization term to the 2D prediction CNN. [31] learns a CNN which predicts an arrangement of cuboids which fits the input voxelic model. These works exploit voxelic models, with a resolution limited to 32^3. Other works have tried to extend 3D learning to more intrinsic representations, such as point clouds. [15] learns a probabilistic graphical model, based on point clouds in correspondence. However, all described methods assume the shapes of the input dataset to be co-aligned, or augment the dataset to learn this variability.

Invariance in deep learning. Our work is also closely related to the study of invariance in deep neural networks, an important topic in deep learning. Usual pooling units in CNNs allow local translation invariance, but deeper work has been done since to achieve better invariance properties. [16] proposes to learn a CNN locally invariant to scale changes, by applying the same convolution at different scales and aggregating the results with a max pooling unit. [11] extends this idea by introducing deep symmetry networks. This work generalizes the invariance property of pooling units to any symmetry group, but still needs to discretize the generating set of the group. Global invariance can be achieved using a global aggregation layer, as explained in [9, 22]. All the considered transformations are passed to the same network, and then aggregated using max pooling. The same technique was already used in [28] to learn rotation-invariant features for 3D recognition. [22] points out that the global aggregation enjoins the network to learn a canonical instance position of the inputs.

Our work generalizes the idea developed in [22] to manifold learning, especially in the 3D context. It learns a more representative embedding of the intrinsic geometry than vanilla autoencoders when the shapes in the dataset are not co-aligned. It also naturally leads to a co-alignment algorithm able to leverage the whole dataset at once. Finally, our framework is generic enough to handle a larger class of invariance group than just translation and rotation groups, or discrete symmetry groups.

3. Quotient autoencoder (QAE)

3.1. Autoencoder limitations

Let $\mathcal{D} = \{x_i \in X \mid i \in [\![1, n]\!]\}$ be a dataset of n shapes representing objects of a same class. Each shape x_i is represented as a vector in a m-dimensional vector space X. In our work, X represents shapes as 3D binary occupancy voxels grids or 2D depth maps, but the idea is generic and could be extended to point clouds for example following [10, 26].

We define the encoder $f_w \colon \mathbb{R}^m \to \mathbb{R}^p$ and the decoder

$g_{w'} : \mathbb{R}^p \to \mathbb{R}^m$, parameterized by weights w and w', where $p \ll m$ is the dimension of the latent space. Encoder and decoder are usually deep feedforward neural networks. The vanilla autoencoder (AE) $g_{w'} \circ f_w$ is classically trained by optimizing a reconstruction error

$$E_{ae}(w, w') = \sum_{i=1}^{n} d(x_i, g_{w'} \circ f_w(x_i)) \,, \tag{1}$$

where $d(x, y)$ is a loss function, comparing the input sample and the sample reconstructed by the autoencoder, such as the usual squared L^2 loss $\|x - y\|_2^2$.

One of the main issues when applying AEs for learning 3D shape representations is that the shapes are not necessarily co-aligned in a common frame. Shapes can have arbitrary poses, as we do not make any assumption on the frame of each shape in the dataset. An option consists in pre-processing the dataset to align the 3D models. However, state-of-the-art automatic co-alignment algorithms are prone to mistakes, as mentioned in section 2. Another option is to directly learn AEs on non-aligned models. To model pose variability, vanilla AEs rely on brute force data augmentation techniques, which consist in applying random transformations to each input sample. The main drawback with data augmentation is that it decreases the modeling power of the network, leading to a decreased performance in terms of reconstruction quality. Indeed, the latent space is less expressive w.r.t. the geometry of the shapes when they are not aligned, as the AE has to encode the orientation variability in addition to the intrinsic shape variability.

3.2. QAE overview

To overcome these limitations, we introduce a new model for unsupervised learning of 3D shapes denoted as quotient autoencoder (QAE). The overall architecture of the proposed QAE scheme is shown in Figure 2. The core idea is to augment the autoencoder with a new quotient loss bringing invariance w.r.t. to a group of transformations, *e.g.* rotations (subsection 3.3). To fully learn in the quotient space and be transformation-invariant we add an orbit pooling layer to the encoder (subsection 3.4). Finally, we derive a training scheme based on error backpropagation to learn all network parameters, which is applicable to both discrete and continuous transformations groups (subsection 3.5).

Formally, let $\mathcal{G}$ be a group, *e.g.* the group of rotations. $\mathcal{G}$ is naturally acting on X by the group action $(h, x) \mapsto h.x$, with $h \in \mathcal{G}$ and $x \in X$, where $h.x$ is the rotation h applied to the shape x. We learn an autoencoder in the quotient space $X/\mathcal{G}$, namely the manifold of the geometries of shapes, independently of their orientation. Thus, the latent space of the QAE is entirely dedicated to the embedding of the intrinsic geometry, leading to a better reconstruction. The intrinsic geometry of a shape x is exactly its orbit $\overline{x}$ under the group action: $\overline{x} = \{h.x \mid h \in \mathcal{G}\}$ (two identical shapes with different orientations have the same orbit, the orbit loses the pose information but keeps all geometric information). Our QAE is able to learn on the quotient dataset $\overline{\mathcal{D}} = \{\overline{x_i} \in X \mid i \in [\![1, n]\!]\}$ itself. We extend our framework to non-rigid transformations in subsection 5.1.

3.3. Quotient loss

We modify the usual autoencoder model in order to directly learn in a quotient space, as described previously. The main idea of the QAE is to replace the loss d of the Equation (1) with its induced quotient loss

$$\overline{d}(x, y) = \inf_{h \in \mathcal{G}} d(h.x, y) \,, \tag{2}$$

which gives the new QAE reconstruction error

$$E_{qae}(w, w') = \sum_{i=1}^{n} \inf_{h \in \mathcal{G}} d(h.x_i, g_{w'} \circ f_w(x_i)) \,. \tag{3}$$

The infimum appearing in the reconstruction error in Equation (3) allows the QAE to reconstruct the input up to any transformation from $\mathcal{G}$ (training part in Figure 2), and thus to focus only on the geometry unlike a vanilla AE. We expect the decoder to be more efficient if it can align all the outputs, since it can dedicate all its capacity to reconstruct similar spatial features at the same place regardless of the input's orientation. Since the loss of the QAE is invariant by any transformation of $\mathcal{G}$ applied to the input, the error of the QAE does not take into account the orientation of the output. Thus, the best way for the decoder to exploit all its capacity is to specialize all its neurons to reconstruct the shapes in a common frame. Let's take the example of a dataset of chairs. It is easier for the decoder to reconstruct the legs of each chair in the same region of the output space, and similarly for all other parts. Of course, the output frame of the QAE has no reason to be a canonical frame (legs at the bottom, back at the top). The co-alignment capacity of the QAE is experimented in subsection 4.3.

In the case where the actions of $\mathcal{G}$ are isometries for a distance d (like the action of rotations), $\overline{d}(x, y)$ is also equal to $\inf_{h \in \mathcal{G}} d(x, h.y)$ and then $\overline{d}$ defines a distance on $X/\mathcal{G}$ (see Proposition 1 in the supplementary material for a proof and [25] for an introduction to quotient topology).

3.4. Orbit pooling

In the QAE, the latent features $f_w(h.x)$ and $f_w(h'.x)$ of two inputs $h.x$ and $h'.x$ differing only by a transformation of $\mathcal{G}$ are likely to be different (unless the QAE perfectly learns such invariance by itself). We propose to explicitly support the latent vector invariance w.r.t. $\mathcal{G}$, in order to have a full quotient architecture additionally to the quotient loss.

To this end, we add an aggregation layer, that we call orbit pooling, at the end of the encoder f of our network

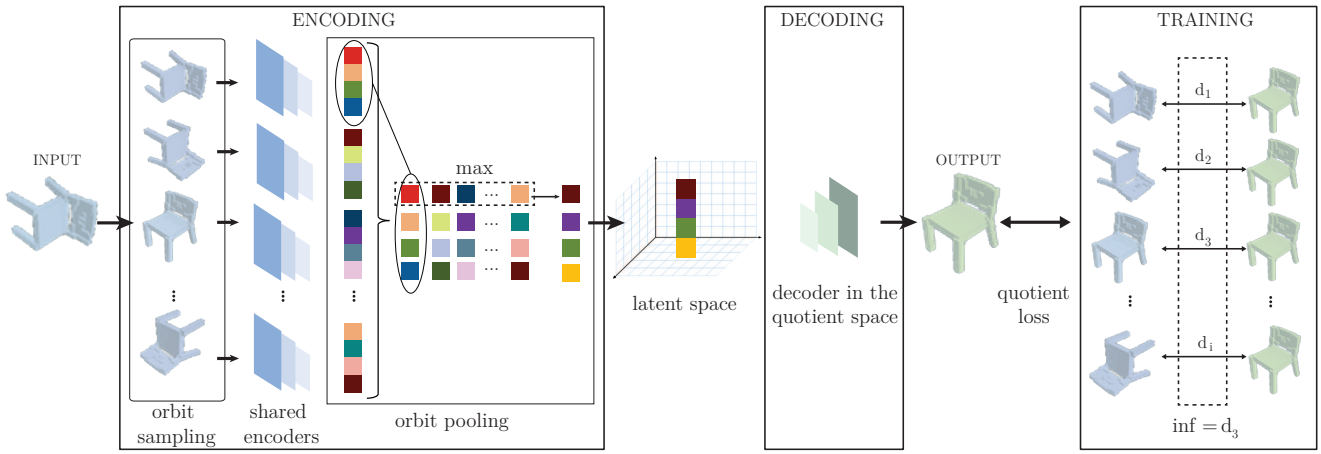

Figure 2: The QAE takes as input a shape and samples its orbit in order to aggregate the latent vectors of each transformed input via the orbit pooling. The encoders share their weights. The quotient loss is the infimum over the distances of the decoder output to each shape of the input orbit. In the figure d_3 achieves this minimum. This new architecture can still be trained with backpropagation.

(see Figure 2). That is, for each input x of the QAE, we randomly sample several transformations $h_j \in \mathcal{G}$ (or all the transformations if $\mathcal{G}$ is finite), and we pass all the transformed inputs $h_j.x_i$ to the same shared encoders f (blue networks in Figure 2), and for each latent vector $z_j = f(h_j.x_i)$ we build the aggregated latent vector z by taking the element-wise maximum[2] along each dimension, that is $z_l = \max_j (z_j)_l$.

This process, already studied in the context of supervised learning in [22, 28], allows the latent vector to capture the shape x modulo the transformation group $\mathcal{G}$, and thus to be much more robust. Indeed, $f_w(x_i)$ now depends only on $\overline{x_i}$ and not the full input x_i itself, that is it depends on x_i independently of the transformations of $\mathcal{G}$, namely independently of the original frame of x_i. Thus, the energy can now be elegantly rewritten wholly in the quotient space [3] as

$$E_{qae}(w, w') = \sum_{i=1}^{n} \inf_{y \in \overline{x_i}} d(y, g_{w'} \circ f_w(\overline{x_i})) \,, \qquad (4)$$

which shows that it depends solely on $\overline{x_i}$ and not x_i. As a result the QAE learns directly on the quotient dataset $\overline{D}$ itself, namely in the quotient space $X/\mathcal{G}$. Our orbit pooling provides a global invariance w.r.t. to any group $\mathcal{G}$, as it travels the orbit of the shape to pool a global invariant feature, which is different from the usual max-pooling units in CNNs that provide only local invariance for translations.

Although our QAE encoder now requires to sample the orbit, the decoder only takes one aggregated latent vector for the whole orbit from one input. Thus, our QAE is effectively twice faster than a vanilla AE to process all augmented data. Overall, the QAE is not necessarily faster to train (see subsection 4.2), because gradient updates are more frequent with a vanilla AE, and the QAE loss landscape is different.

3.5. QAE training

As detailed in section 4, we rely on deep convolutional neural networks (ConvNets) for implementing QAE encoder and decoder. Thus, we aim at deriving a continuous optimization scheme based on error backpropagation for efficiently training all network parameters (w, w'). In this context, the main challenge is to be able to compute the gradient $\nabla_{w,w'} E_{qae}(w, w')$ in Equation (3).

The straightforward strategy we adopt in the first place is to discretize $\mathcal{G}$ into a finite group $\mathcal{G}'$ (which is a subgroup of $\mathcal{G}$), so that the infimum becomes actually a minimum. If we replace $\mathcal{G}$ with the finite subgroup $\mathcal{G}' = \{h_1, \cdots, h_k\}$, the loss becomes

$$\min_{h \in \mathcal{G}'} d(h.x_i, g_{w'} \circ f_w(x_i)) \,. \qquad (5)$$

The minimum in Equation (5) is differentiable almost everywhere w.r.t. (w, w'), which leads to a direct application of backpropagation for training this discrete QAE version.

In the more general continuous case, we assume that $\mathcal{G}$ is a Lie group, parameterized by a continuous finite-dimensional vector h. Under mild assumptions on the group $\mathcal{G}$ (for instance if $\mathcal{G}$ is compact and the group action is continuous), the infimum in Equation (3) is actually a minimum, reached in $\widehat{h_x}(w, w')$, that we can compute with gradient descent or Gauss-Newton (comparisons are made in subsection 5.2). However, the computation of the gradient of the quotient loss becomes more challenging. If

[2] An aggregation function of L latent vectors of dimension p is a function $\delta \colon (\mathbb{R}^p)^L \to \mathbb{R}^p$ whose main properties are that δ is invariant under permutation, and $\delta(h, h, , h) = h$ for any $h \in \mathbb{R}^p$.

[3] Notice that the loss in Equation (2) $\inf_{h \in \mathcal{G}} d(h.x_i, g_{w'} \circ f_w(x_i))$ can actually be rewritten $\inf_{y \in \overline{x_i}} d(y, g_{w'} \circ f_w(x_i))$.

we define $\mathcal{L}_x\colon ((w,w'),h) \mapsto d(h.x, g_{w'} \circ f_w(x))$, and $\widehat{\mathcal{L}_x}\colon (w,w') \mapsto \mathcal{L}_x((w,w'),\widehat{h_x}(w,w'))$, then

$$\nabla_{w,w'}\, \overline{d}(x, g_{w'} \circ f_w(x)) = \nabla\, \widehat{\mathcal{L}_x}(w,w')\,. \qquad (6)$$

A priori it would require to compute the derivatives of $\widehat{h_x}$ w.r.t. (w,w'). Indeed, unlike the discrete case, even an infinitesimal change in (w,w') modifies $\widehat{h_x}$, whereas it is constant almost everywhere in the discrete case.

Fortunately, we can take advantage of the envelope theorem [24] to simplify the gradient computation. Indeed, we have $\nabla_h\, \mathcal{L}_x((w,w'),\widehat{h}(w,w')) = 0$ by definition of $\widehat{h}$, so

$$\begin{aligned}\nabla\, \widehat{\mathcal{L}_x}(w,w') &= \nabla_{w,w'}\, \mathcal{L}_x((w,w'),\widehat{h}(w,w')) \\ &\quad + J_{\widehat{h_x}}(w,w')^{\mathrm{T}}\, \nabla_h\, \mathcal{L}_x((w,w'),\widehat{h}(w,w')) \\ &= \nabla_{w,w'}\, \mathcal{L}_x((w,w'),\widehat{h}(w,w'))\,. \qquad (7)\end{aligned}$$

The intuition behind this simplification is that the second-order terms which vanish in the gradient computation make it possible to consider $\widehat{h}$ constant during the backpropagation, as in the discrete case.

3.6. Semi-supervised learning

If a small subset of co-aligned shapes is available, we can leverage it in order to reinforce the co-alignment effect of the QAE. We can also use this co-aligned dataset to force the alignment in its orientation, instead of letting the QAE co-align in an arbitrary (random) position. To achieve this goal, we first pre-train the chosen architecture of the QAE (including the orbit pooling layer) with a regular non-quotient loss (*i.e.* the loss of a vanilla autoencoder in Equation (1)). Then, we fine-tune the QAE architecture on the remaining misaligned dataset using the quotient loss. We validate the significant boost in terms of co-alignment performances of this strategy in subsection 4.3.

4. Experiments

4.1. Experimental setup

We evaluate our model on the ShapeNet dataset [6], which contains a few thousands 3D models for several categories. We focus on chairs, planes, and cars, as these categories exhibit complex geometries and are of primary interest for the CAD industry. Each 3D model is represented as a mesh in the dataset, aligned in a common frame for each category. For each mesh we compute a 2D depth map (in a representative view) of size 64×64 and a 3D volumetric occupancy binary grid of size 32^3, following a standard protocol (*e.g.* [8,36]). We experiment the QAE on both 2D depth maps and 3D voxels grids to illustrate its performances.

We explore the discrete QAE with the rotations (continuous QAE is experimented in subsection 5.2). We train the 2D QAE with a discretized subgroup of $SO(2)$ made of 36 rotations . Besides, CAD models are generally designed such that their 3 main directions are aligned with the frame's axes, but we do not know which direction corresponds to x, y, or z axis, as it depends on each 3D model and the conventions used by the designer. So we decided to experiment the 3D QAE not on $SO(3)$ but on the group of the 24 rotational symmetries which leave the cube unchanged (the octahedral group), in order to be representative of a practical use case. Indeed, we also prove the relevance of the approach with chairs extracted from the SketchUp 3D Warehouse [1]. These chairs are designed in different frames, but all these frames can be aligned up to a transformation from the octahedral group, showing that real datasets can be representative of our hypothesis. Both 2D and 3D datasets have been misaligned, in order to compare with the vanilla autoencoder. To be consistent the misalignment in 3D is done by applying a random rotation of the octahedral group.

We build a 2D and a 3D deep convolutional architecture for the QAE, detailed in the supplementary material. We compare with vanilla autoencoders with the same architectures, except that there is no orbit pooling. As both groups are finite, we can provide the full orbit of rotated shapes to the orbit pooling layer instead of doing a random partial orbit pooling. We adopt the usual squared L^2 loss for the distance d, and train all the autoencoders as denoising autoencoders (with a classical independent Bernoulli noise layer at the input of the encoder, see [32]). We split each category into training, evaluation and test datasets containing respectively 80%, 15% and 5% of the samples. In all experiments we use ADAM optimization [19].

4.2. Reconstruction and quantitative experiments

	planes		*chairs*		*cars*	
	2D	3D	2D	3D	2D	3D
AE 1	0.14	0.48	0.32	2.2	0.56	0.67
AE 2	0.12	0.44	0.28	2.0	0.52	0.60
QAE	**0.10**	**0.36**	**0.22**	**1.9**	**0.45**	**0.48**

Table 1: Reconstruction errors on each 2D (depth maps) and 3D (voxels) test dataset for the vanilla autoencoder without data augmentation (AE 1), with data augmentation on rotations (AE 2), and the QAE with the pooling of all rotated inputs (QAE).

The first experiment is dedicated to the comparison of our QAE versus a vanilla autoencoder. We train the QAE on the three misaligned datasets, both in 2D and 3D. We also train two vanilla autoencoders with the same architecture than the QAE, one with the same dataset as for the QAE, and another with an augmented dataset including all allowed rotations for each shape in order to let the vanilla autoencoder learn by itself the rotation variability.

The results are summarized in Table 1. These results

prove that the QAE achieves a lower reconstruction error compared to a vanilla autoencoder, even with data augmentation (AE2 in the table). Indeed, the latent space being dedicated to model the intrinsic geometric variability of the data, the capacity of the QAE is larger than the vanilla one. Our QAE beats the vanilla autoencoder on all three datasets both in 2D and 3D, mostly with a significant margin up to a 20% decrease in the reconstruction error relative to AE2 (*e.g.* 3D cars, 3D planes, 2D chairs). The QAE is just slightly better only on the 3D chairs dataset. We mainly explain such a difference by the nature of the dataset, since 3D chairs have a very complex structure harder to learn, as proved by the high reconstruction error for both the QAE and the autoencoder.

Figure 3 evinces that the lower reconstruction error of the QAE effectively corresponds to a visually better reconstruction. Indeed, the reconstructions from the vanilla autoencoder do not exhibit a sharp geometry, such as wheels or body, unlike the QAE which provides a more detailed shape. Notice that the QAE does not reconstruct the shape in its original frame, but co-aligns the shapes. This side effect, raised in subsection 3.3, is studied in subsection 4.3.

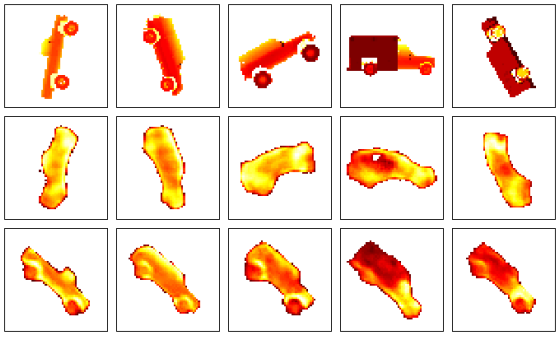

Figure 3: The first row shows the original depth maps, the second row the reconstruction with the vanilla autoencoder trained with data augmentation, the third row the reconstruction with the QAE.

We also experiment the influence of the orbit pooling on the learning performance. Figure 4 shows the different training curves on the chairs depth maps, and validates the introduction of the orbit pooling explained in subsection 3.4. Indeed, we observe that without the orbit pooling layer (blue curves) the QAE does not beat the vanilla autoencoder (black curves). Nevertheless, with a full pooling (red curves) the QAE reaches a much smaller reconstruction error (0.21 vs 0.28 on the validation set after 200 epochs). Moreover, the partial orbit pooling strategy, instead of pooling over all the transformations of the discretized group, is also validated by the plots (green curves, loss of 0.24 on the validation set after 200 epochs). With this partial pooling we achieve a substantial faster training than with the full pooling (200 epochs take 2h01m in the first case vs 54m in the latter) while still getting a reconstruction error far below the vanilla autoencoder with data augmentation.

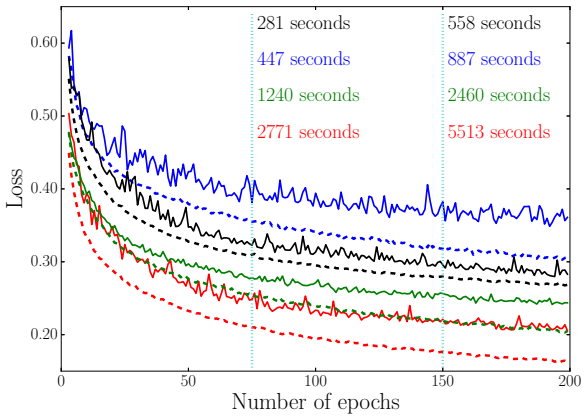

Figure 4: The dashed (resp. solid) curves represent the training (resp. validation) loss. Black: vanilla autoencoder with data augmentation in rotations. Red: QAE with full pooling of all 36 rotated inputs. Green: QAE with partial pooling of 12 random rotated inputs. Blue: QAE without orbit pooling. Vertical lines show elapsed training times for each case.

4.3. Shapes co-alignment

The QAE is also able to reconstruct most of the shapes in a common frame, as illustrated in Figure 5 for the 2D QAE and in Figure 6 for the 3D QAE, providing a new co-alignment algorithm with several advantages. It can deal with shapes exhibiting a large geometric variability, and is able to automatically co-align any new data on the fly in real-time once the QAE has been learned. For each input x, we simply replace x with $\widehat{h_x}.x$ where $\widehat{h_x} = \arg\min_{h\in\mathcal{G}} d(h.x, g \circ f(x))$.

For the depth maps datasets, all chairs have been correctly co-aligned in the same frame by the QAE, just as cars (see Table 2). In 3D the diversity of the shapes does not allow the QAE to naturally co-align all the data in a single orientation. Among the 24 possible rotations, two are used by the QAE to reconstruct the input shape. The QAE internally clusters the dataset in two parts, which leads to two different orientations for each cluster (see Figure 6). It is not easy to see a clear semantic meaning of each cluster for the chairs, but very clearly the QAE has learned on the planes dataset to separate the aircrafts with low aspect ratio[4] from the aircraft with high aspect ratio (see second row in Figure 6), without any error.

As explained in subsection 3.6, we can use a small pre-processed dataset with aligned shapes in order to constrain the QAE and improve the co-alignment performance. We pre-train the QAE with an aligned dataset containing 5% of the total number of shapes of the whole training dataset, and we fine-tune with the remaining non-aligned

[4]The aspect ratio of a wing is the ratio of its span to its mean chord.

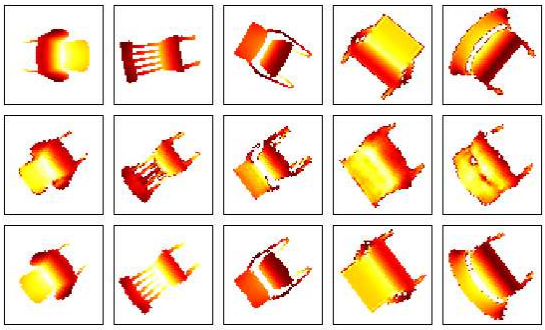

Figure 5: The first row shows input chairs, the second row the chairs reconstructed by the QAE, the third the original input chairs with the orientation given by the QAE. Last two columns represent specifically generated wider chairs, very different from the training dataset, to illustrate the robustness of the QAE.

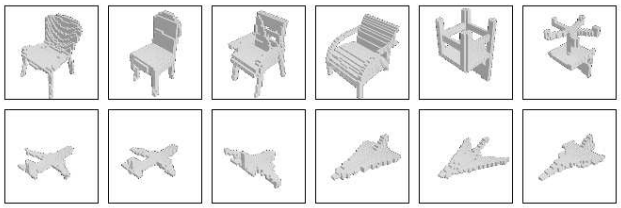

Figure 6: Orientations given by the QAE applied to chairs and planes. The QAE automatically co-aligned on 2 orientations among 24, depending on an internal clustering learned by the QAE.

		w/o supervision	*semi-supervised*
chairs	2D	0.0%	0.0%
	3D	20%	1.8%
cars	2D	0.0%	0.0%
	3D	13%	2.8%
planes	2D	31%	0.0%
	3D	28%	6.8%

Table 2: Co-alignment errors for the QAE trained on a completely misaligned dataset, and the co-alignment results for the QAE pre-trained on a small aligned dataset. The alignment error is given by one minus the greatest number of co-aligned shapes over the total number of shapes, on the misaligned test set.

95% shapes. Results are summarized in Table 2, we see that the semi-supervised approach yields a great improvement in co-alignment results. 3D chairs and cars are almost all perfectly co-aligned. Results are a bit inferior with planes, as it is a smaller dataset with more geometric ambiguities, especially with a resolution of 32^3.

Besides, the experiment shown in Figure 5 also proves that the QAE does more than a simple PCA alignment, since it is able to co-align chairs which are wider than high (last two columns), proving that the QAE has really learned a hidden structure specific to the chairs. We point out that these uncommon chairs have been specifically generated to test the robustness of our approach, and thus are not representative of the initial dataset. Although the QAE has never seen wide chairs it is still able to co-align them.

Figure 7 demonstrates an application of our 3D co-alignment to a practical case. The first row shows that different chairs from the 3D Warehouse of SketchUp are not necessarily designed in a common frame, but they are all designed so that their main axes are aligned with the x, y, z axes. Thus, there are 24 possible design conventions. Our algorithm allows to straighten all theses meshes in a common frame. To find the transformation needed to co-align each mesh, we voxelize them and co-align the voxelizations with the 3D QAE.

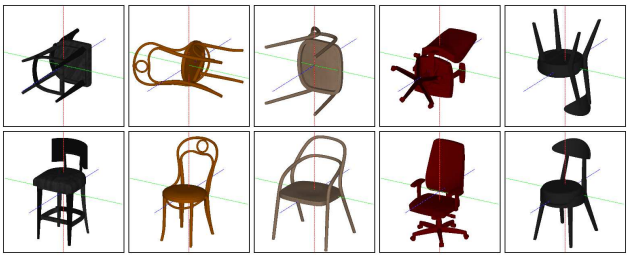

Figure 7: The first row shows meshes from SketchUp 3D Warehouse in their original frame, the second row shows the same meshes straightened with our 3D QAE.

An interesting application of the co-alignment provided by our QAE is the possibility to interpolate the geometry of two shapes which are originally misaligned, as illustrated in Figure 8. Shapes interpolation is done by a linear interpolation of the corresponding latent vectors, and thus the interpolation is done non-linearly in the original space.

The classical interpolation with a vanilla autoencoder, despite having been trained with shapes with arbitrary orientations, is not able to disentangle the pose from the geometry. Instead of rotating the chair and interpolating the geometry, we see that the interpolation seems to perform a kind of optimal transport of one depth map to the other. As a result the interpolated chairs are not realistic. On the contrary, the interpolation in the latent space of the QAE interpolates the geometry independently of the orientation of the initial shapes, leading to realistic interpolations of chairs.

5. Quotient by non-rigid deformations

5.1. Extension of the QAE

The QAE framework we described in section 3 can be applied to a set of non-rigid transformations $\mathcal{G}$, without any modification. It is not even compulsory for $\mathcal{G}$ to be a group, but then we may lose some important properties (for instance $X/\mathcal{G}$ is not a quotient space anymore).

We can parametrize non-rigid transformations by RBF

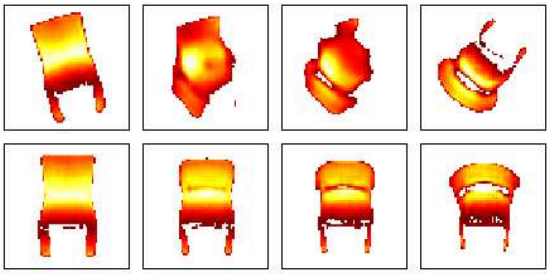

Figure 8: The first row shows several interpolations between two chairs with a vanilla autoencoder trained with data augmentation. The second row displays the same interpolation with the QAE.

(radial basis function) interpolation to warp images or voxels grids, in order to non-rigidly align the shapes. In the 2D case we set 5 control points $\{c_1, \dots, c_5\}$ at the same fixed positions on each input image. Denoting ϕ the RBF, our transformations set is made of the operators $h(a), a \in \mathbb{R}^{10}$, such that $(h(a).I)(x,y) = I(x + \sum_{i=1}^{5} a_i \phi(\|c_i - (x,y)\|_2, y + \sum_{i=1}^{5} a_{2i} \phi(\|c_i - (x,y)\|)$. We use the thin plate spline for ϕ [5].

5.2. Experiment on feature matching

We aim to learn a reference space where all the shapes share a common representation. Once the data is aligned in the reconstructed space, we can match the input shapes by morphing the reconstructed shapes to the original ones. This allows to backpropagate features for instance, as experimented now on chairs depth maps.

We train the QAE on our RBF-based continuous set of non-rigid transformations, with an orbit pooling of 12 random transformed inputs. The optimization of the infimum appearing in the Equation (2) is done with Gauss-Newton, where the normal equation is solved with conjugate gradient. We compare it to a basic gradient descent in Figure 9, showing that Gauss-Newton optimization is more efficient (0.41 vs 0.60 for the validation loss). Since depth is less relevant here, training was done on occupancy masks in order to optimize the learning.

Figure 10 shows how we can match features across different shapes by retargeting common points on the learned reference space. Reconstructed shapes (first row) do not look like plausible chairs, as it is not an objective in this experiment, the main goal being to match the shapes in a common space. To perform feature matching we pick a set of points (green marks in the bottom left chair) and warp them in the corresponding reconstructed shape. We copy these warped points across all the reconstructed shapes (red marks of the first row, they all have the same coordinates over the different shapes) and retarget them into the original input shapes (blue marks of the second row).

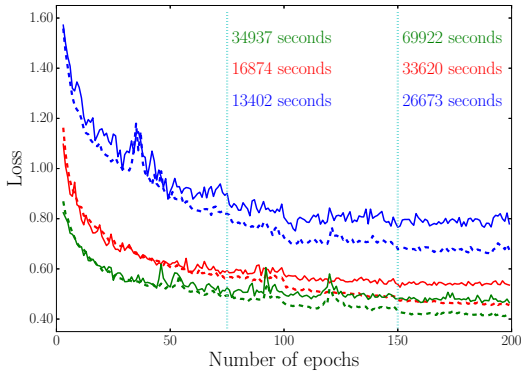

Figure 9: The dashed (resp. solid) curves represent the training (resp. validation) loss for the continuous QAE. The blue curves display the QAE loss where $\widehat{h}(w)$ is obtained with a gradient descent with 25 iterations, and the red (resp. green) curves display the QAE loss where $\widehat{h}(w)$ is computed with 10 (resp. 15) iterations of Gauss-Newton, each one being solved with 5 (resp. 10) iterations of conjugate gradient.

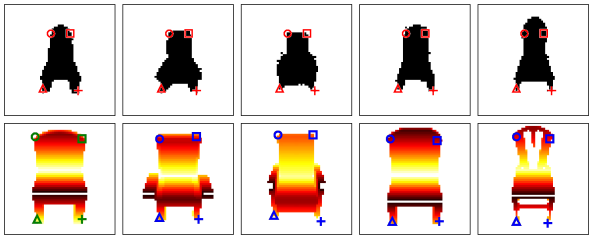

Figure 10: All the red marks have the same coordinates. They are retargeted on each original chair to match the green marks with each blue mark on the second row. Retargeting of feature points is shown on the original depth maps.

6. Conclusion

In this work we introduce a new deep autoencoder model to learn a quotient manifold, in a fully unsupervised framework. Our QAE encompasses a new quotient loss, and an orbit pooling, in order to learn in the quotient space itself. We also explain how this new architecture can be trained for both discrete and continuous transformations. We show that our autoencoder quotiented by rotations leads to a better representation of the intrinsic geometry. Especially, we provide quantitative results proving that the QAE is more efficient than a vanilla autoencoder trained with data augmentation. We also illustrate the ability of the QAE to automatically co-align shapes, leading to other interesting applications such as geometric interpolations of misaligned shapes. We finally extend our model to non-rigid transformations, as illustrated by our feature matching experiment.

References

[1] Google 3d warehouse. https://3dwarehouse.sketchup.com. 5

[2] M. Averkiou, V. G. Kim, and N. J. Mitra. Autocorrelation descriptor for efficient co-alignment of 3d shape collections. *Computer Graphics Forum*, 35:261–271, 2015. 2

[3] Y. Bengio, L. Yao, G. Alain, and P. Vincent. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems*, volume 26, pages 899–907. 2013. 2

[4] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006. 1

[5] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567–585, June 1989. 8

[6] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 5

[7] M. Chaouch and A. Verroust-Blondet. Alignment of 3d models. *Graphical Models*, 71(2):63–76, Mar. 2009. 2

[8] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European Conference on Computer Vision*, pages 628–644, 2016. 5

[9] S. Dieleman, J. De Fauw, and K. Kavukcuoglu. Exploiting cyclic symmetry in convolutional neural networks. In *International Conference on Machine Learning*, volume 48, pages 1889–1898, 2016. 2

[10] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *Computer Vision and Pattern Regognition*, 2017. 2

[11] R. Gens and P. M. Domingos. Deep symmetry networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27, pages 2537–2545. 2014. 2

[12] R. Girdhar, D. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, 2016. 2

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, pages 2672–2680. 2014. 2

[14] G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504 – 507, 2006. 2

[15] H. Huang, E. Kalogerakis, and B. Marlin. Analysis and synthesis of 3d shape families via deep-learned generative models of surfaces. *Computer Graphics Forum*, 34(5), 2015. 2

[16] A. Kanazawa, A. Sharma, and D. W. Jacobs. Locally scale-invariant convolutional neural networks. In *Deep Learning and Representation Learning Workshop: NIPS*, 2014. 2

[17] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Eurographics Symposium on Geometry Processing*, pages 156–164, 2003. 1

[18] D. G. Kendall. A survey of the statistical theory of shape. *Statistical Science*, 4(2):87–99, 05 1989. 1

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, volume 3, 2015. 5

[20] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, volume 2, 2014. 2

[21] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009. 1

[22] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys. TI-POOLING: transformation-invariant pooling for feature learning in convolutional neural networks. In *Computer Vision and Pattern Recognition*, pages 289–297, 2016. 2, 4

[23] R. Litman and A. M. Bronstein. Learning spectral descriptors for deformable shape correspondence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(1):171–180, Jan. 2014. 1

[24] P. Milgrom and I. Segal. Envelope theorems for arbitrary choice sets. *Econometrica*, 70(2):583–601, 2002. 5

[25] J. Munkres. *Topology*. Featured Titles for Topology Series. Prentice Hall, 2000. 3

[26] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Computer Vision and Pattern Regognition*, 2017. 2

[27] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, volume 32, pages 1278–1286, 2014. 2

[28] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *International Conference on Computer Vision*, 2015. 2, 4

[29] J. Sun, M. Ovsjanikov, and L. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Symposium on Geometry Processing*, pages 1383–1392, 2009. 1

[30] G. K. Tam, Z.-Q. Cheng, Y.-K. Lai, F. Langbein, Y. Liu, A. D. Marshall, R. Martin, X. Sun, and P. Rosin. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. *IEEE Transactions on Visualization and Computer Graphics*, 19(7):1199–1217, July 2013. 2

[31] S. Tulsiani, H. Su, L. J. Guibas, A. A. Efros, and J. Malik. Learning shape abstractions by assembling volumetric primitives. In *Computer Vision and Pattern Regognition*, 2017. 2

[32] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine Learning*, pages 1096–1103, 2008. 5

[33] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371–3408, Dec. 2010. 2

[34] Y. Wang, S. Asafi, O. van Kaick, H. Zhang, D. Cohen-Or, and B. Chen. Active co-analysis of a set of shapes. *ACM Transactions on Graphics*, 31(6):165:1–165:10, Nov. 2012. 2

[35] J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. In *Advances In Neural Information Processing Systems*, volume 29, pages 82–90, 2016. 2

[36] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Computer Vision and Pattern Recognition*, pages 1912–1920, 2015. 2, 5