Eurographics Symposium on Geometry Processing 2017
J. A. Bærentzen and K. Hildebrandt
(Guest Editors)

Volume 36 (2017), Number 5

The shape variational autoencoder: A deep generative model of part-segmented 3D objects

C. Nash[1] and C. K. I. Williams [1,2]

[1]University of Edinburgh, UK
[2]Alan Turing Institute, UK

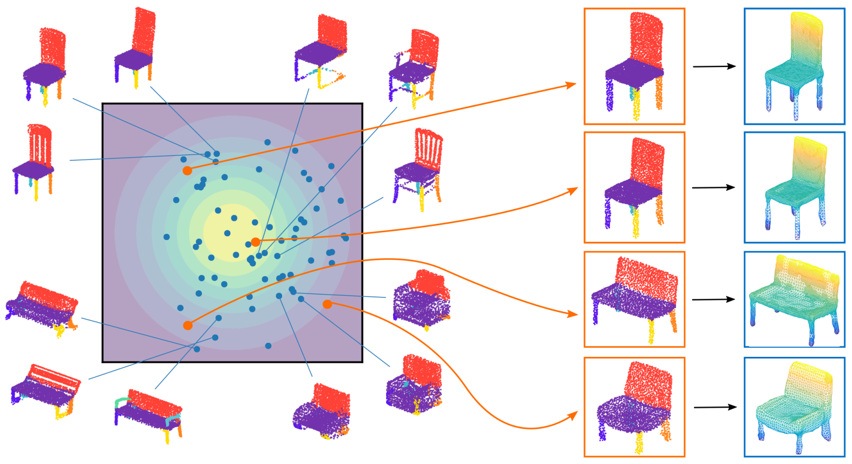


Figure 1: **Data driven synthesis of 3D objects.** (Left) Given an input collection of oriented surface points from an object class we learn a low-dimensional shape embedding using a deep probabilistic auto-encoder. (Middle) Samples from the prior embedding distribution can be passed through the decoder to obtain novel shape examples complete with point orientations (not shown). (Right) Smooth meshes are constructed by making use of the sampled point orientations.

Abstract

We introduce a generative model of part-segmented 3D objects: the shape variational auto-encoder (ShapeVAE). The ShapeVAE describes a joint distribution over the existence of object parts, the locations of a dense set of surface points, and over surface normals associated with these points. Our model makes use of a deep encoder-decoder architecture that leverages the part-decomposability of 3D objects to embed high-dimensional shape representations and sample novel instances. Given an input collection of part-segmented objects with dense point correspondences the ShapeVAE is capable of synthesizing novel, realistic shapes, and by performing conditional inference enables imputation of missing parts or surface normals. In addition, by generating both points and surface normals, our model allows for the use of powerful surface-reconstruction methods for mesh synthesis. We provide a quantitative evaluation of the ShapeVAE on shape-completion and test-set log-likelihood tasks and demonstrate that the model performs favourably against strong baselines. We demonstrate qualitatively that the ShapeVAE produces plausible shape samples, and that it captures a semantically meaningful shape-embedding. In addition we show that the ShapeVAE facilitates mesh reconstruction by sampling consistent surface normals.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling—

1. Introduction

The computer graphics industry relies to a large extent on the 3D content created by artists, modellers, designers and animators. The

Accepted version of the paper to appear in Computer Graphics Forum 36(5), presented at the Symposium on Geometry Processing, July 2017

content creation process is time consuming and intensive even for highly skilled graphics artists. 3D Content is often created from scratch, despite the fact that vast collections of 3D models exist in online repositories and private collections. These shape collections contain considerable information about the styles, structures and textures of object classes. It is desirable to leverage this information with tools that can aid designers in the modeling process. Such tools can help by enforcing data-driven constraints, providing completions of partially designed objects, or even through the synthesis of whole shapes.

The ability to automatically synthesize and analyze 3D objects is useful not only for graphics applications, but in computer vision, where there has been a recent focus on the use of 3D shape representations in scene understanding tasks; see [ZSSS13, SHM*14, CSCS15]. A detailed representation of object shape allows for complex 3D reasoning, and a model of shape variability aids the performance of recognition tasks in images. Structures such as 3D bounding boxes [PT11, LS10], wireframe models [ZSSS13], or 3D CAD models [CSCS15] have been used as shape representations and successfully recognized in images.

In this work we present the shape variational auto-encoder (ShapeVAE), a model of structural and local shape variability that captures a distribution over the co-existence of object parts, the locations of 3D surface points, and the surface normals associated with these points. We make use of the representation described by Huang *et al*. consisting of a collection of dense point correspondences, segmented into the object's constituent parts and augment it with point normals [HKM15]. We take a powerful class of deep generative model, the variational autoencoder, and introduce a novel architecture that leverages the hierarchichal part-structure of 3D objects. Our model is capable of generating plausible, novel point cloud objects, and by generating consistent point normals, we can take advantage of powerful surface reconstruction methods to reconstruct smooth mesh geometry. We demonstrate that the ShapeVAE achieves strong performance in a shape completion task in comparison to a linear baseline, while producing samples of a higher quality. We further show that the ShapeVAE learns a semantically meaningful latent space, and that by sampling both point sets and surface normals, the ShapeVAE enables the use of powerful surface reconstruction techniques. In addition the ShapeVAE is considerably more efficient to train compared to related work, and it allows for efficient sampling and missing data imputation.

2. Related work

Our methods relate to statistical models of objects and shapes as well as general deep generative models. We provide an overview of the most relevant prior work.

Shape synthesis and generative shape models. Generative models of 3D objects have been proposed for a range of shape representations including 3D voxel images [WSK*15, GFRG16], keypoints [HKM15], and meshes [KCKK12, ZB15, YAMK15]. Early work includes active shape models [CTCG95]; statistical models of corresponding landmark points that have been used to model face shape [LTC97], medical images [HM09], and cars [ZSSS13]. Such models are often applied to relatively few landmark points, and as such are more useful for analysis rather than synthesis of whole objects.

Much of the recent work on generative shape models has focused on voxel representations of 3D objects. Wu *et al.* model the joint distribution of voxels and object class labels with a convolutional deep belief network [WSK*15]. The authors use the model to recognise object classes and reconstruct 3D objects from a single depth image. Girdhar *et al.* use a 3D convolutional auto-encoder to establish a compressed vector representation of 3D objects that can be predicted and reconstructed in real images [GFRG16]. A 3D generative adversarial network with convolutional structure was used by Wu *et al.* to synthesize voxel objects [WZX*16]. Volumetric representations have the advantage that different objects are directly comparable on the voxel level, whereas triangulated meshes do not have an explicit parameterization that is consistent across instances. However naive volumetric methods are limited in terms of resolution, as the dimensionality is cubic in the width of the voxel grid. In addition, voxel grids require modelling of many redundant dimensions, such as empty space inside or outside the object itself, although recent work using octree representations seeks to address this [TDB17].

Our methods model object surfaces in addition to the structural variability associated with the presence or absence of certain parts. Other work has similarly made use of the part-decomposability of objects in their shape models, either by explicitly recombining part instances from a database [KCKK12, AKZM14, ZCM13], or by incorporating parts as a modelling structure [FAvK*14, ZB15]. Kalogerakis *et al.* learn a generative model over continuous and discrete geometric features of object parts and synthesize shapes by matching the generated features to object parts in database [KCKK12]. Averkiou *et al.* fit templates consisting of deformable cuboids to large collections of 3D objects and obtain a low-dimensional hierarchical embedding, that captures semantic similarity between the input objects [AKZM14]. The authors designed a method for interactive object synthesis in which a user can explore the embedding space, and create new objects by deforming parts from nearby objects. Fish *et al.* also used a parts-based method in which they modelled the geometric relationships between shape parts [FAvK*14]. For each shape, unary relations such as the relative length of a part, and binary relations, e.g. the angle between two parts were captured. In related work Zuffi *et al.* develop a parts-based 'stitched puppet' model of human shape which allows for the shape and pose of body parts to be modelled separately, while encouraging connecting parts to be close together [ZB15]. This allows for shape variation to be captured on various levels: on the global level articulated pose is modelled, and on the local level continuous shape deformation is modelled. Our methods are different in that although we make use of object parts, we also use a highly detailed shape representation consisting of dense point clouds.

Recent work using dense point clouds includes PointNet [QSMG16], in which unordered point sets are processed using deep networks with a symmetric pooling function. Such networks were shown to be effective in semantic segmentation and object classification tasks. However, PointNet is not a generative model of point sets, but rather it maps input point sets to output such as a model classification, or part segmentation. In related work, a conditional generative model of unordered point sets was introduced in [FSG16], where given an image, a collection of 3D output points was synthesized that captures the coarse shape of objects in the image. The closest work to ours is Huang *et al.* in which part-segmented 3D keypoints are modelled with the beta shape machine (BSM), a variant of a multi-layer Boltzmann machine that captures global and local shape variation with a similar part-oriented structure [HKM15]. This model is demonstrated to be effective at generating plausible shapes, as well as for shape segmentation and fine grained classification tasks. Unlike the BSM which is an undirected probabilistic model, the models in this paper are directed, and as such training and sampling is more rapid, and we may more easily scale to high-dimensional data. When we model surface normals as well as points we double the dimensionality of the data-space, and being able to efficiently train in very high-dimensional space becomes important.

Deep generative models. Our generative model of oriented point clouds makes use of a variant of a probabilistic model known as a variational autoencoder (VAE) in the machine learning literature [KW14]. In a VAE a prior distribution is specified over latent variables, and data is generated by mapping the latent variables through a non-linear function implemented by a neural network. Variational autoencoders are used to model images [KW14] and voxel data [BLRW16] and been shown to learn semantically meaningful representations of the data. The variational autoencoder is an example of a deep generative model (DGM): a class of generative model that employs deep neural network architectures.

Other DGMs include deep Boltzmann machines (DBMs) [SH09]. DBMs are undirected models that have been used to capture complex distributions over speech data [MDH12], images [EHWW14, RHSW11] and part-segmented objects [HKM15]. Although DBMs are flexible, they can be difficult and time-consuming to train, and are more complicated to sample from than directed models. Generative adversarial networks (GANs) are a powerful class of DGN in which a generator network maps low-dimensional latent samples to the data space, and a discriminator network is trained to distinguish between real and fake samples [GPM*14]. By training the generator neural network to to fool the discriminator, samples are pushed closer to the true data distribution. GANs have been used to model images [GPM*14, DCSF15] and voxels [WZX*16], and are notable for producing sharp, high-quality samples. However, GANs do not explicitly describe a probability distribution over data and as such are difficult to evaluate.

In this work we take the VAE, a powerful class of deep generative model that enables efficient sampling, density estimation and conditional inference, and introduce structure that captures the hierarchical part-structure of 3D objects.

3. Overview

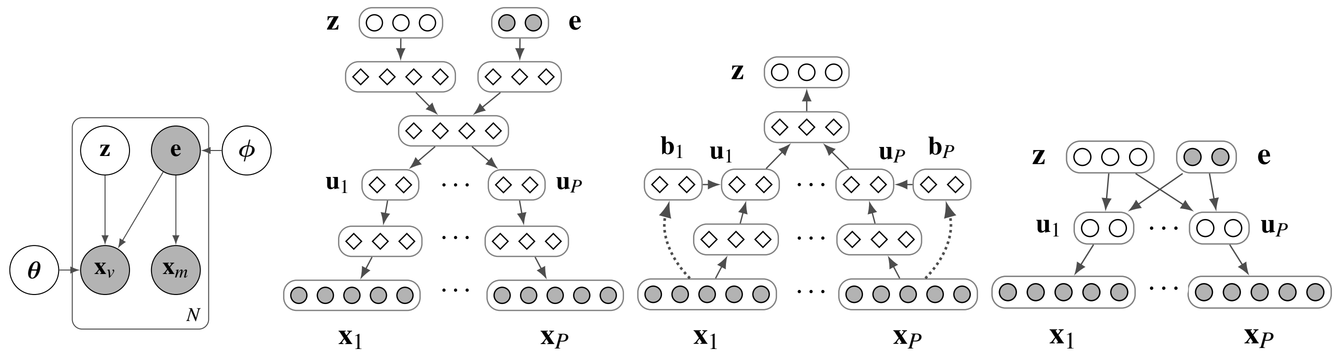
Our goal is to take a collection of segmented input objects with dense point correspondences and to learn a generative model of 3D shape such that we can synthesize novel examples, complete partial objects, and embed 3D objects in a low-dimensional latent space. In this section we provide an overview of our deep generative shape model, a description of the data sets used, and the required pre-processing.

Generative model for oriented point clouds. The core of our method is a generative model that describes a probability distribution over surface points, surface normals, and part existences for large collections of 3D objects. The relationships between these variables in 3D objects is highly complex due to hierarchical part relationships, symmetry relationships, as well as local smoothness and other structural constraints. Our model is a variant of a variational auto-encoder: a powerful generative model capable of capturing complex distributions over high-dimensional data. The VAE consists of an encoder network that maps data to a low-dimensional latent code, and a decoder that maps the latent code to a reconstruction of the the data. In doing so the VAE is forced to make the hidden code highly informative about its associated data. We equip our VAE with a hierarchical architecture in which higher layers capture global, structural relationships in objects, and lower levels capture variability within object parts. After training the ShapeVAE we gain the ability to sample new instances, perform shape completion and a lower bound on the likelihood of unseen instances. We also obtain a compact shape descriptor in the form of the highest level latent variables, and an encoder network that can map a data instance to this high-level description efficiently.

Data and pre-processing. Our methods assume a collection of consistently aligned and scaled 3D shapes for which point-wise correspondences, consistently-oriented surface normals and consistent segmentations are available. We assume that each object class has a fixed number of parts, and that these parts can be present or absent for any particular object example. For example an airplane can have up to 6 parts: the fuselage, two wings, two horizontal tail fins, and one vertical tail fin. In most instances a plane will have all of these parts, but in some cases the vertical of horizontal tail fins may not be present. Figure 6a shows some example planes with part segmentations. We make use of such collections provided by Huang *et al.* [HKM15], and note that there exist effective methods for automatic analysis of 3D object databases that obtain correspondences and segmentations as an output [HKM15, KLM*13]. The datasets we use feature chair and airplane object classes with 3701, and 1509 examples respectively. The 3D meshes were originally collected from the Trimble Warehouse online repository by Kim *et al.* [KLM*13].

4. Generative model for oriented point clouds

In this section we describe our generative model of 3D objects, the shape variational autoencoder. The ShapeVAE aims to model the joint distribution over part existences, surface points and surface normals. This task is made challenging by a number of factors. The dimensionality of the data can be extremely high, with an object class with 5000 surface points having 15000 variables describing point locations, and a further 15000 variables describing surface normals. This poses difficulties for modelling, as even with thousands of data examples, only a tiny portion of this 30000-dimensional space can be covered. Beyond this, there are a range of complex dependencies that must be captured in order to be able to synthesize plausible shapes. The model must capture symmetry,



(a) Graphical model (b) ShapeVAE decoder (c) ShapeVAE encoder (d) ShapeFA

Figure 2: **Generative models of part-segmented shapes:** Filled circles represent visible variables, unfilled circles represent latent variables, and diamonds represent deterministic variables. (a) Latent variables $\mathbf{z}$ are sampled from a prior distribution, and existence variables $\mathbf{e}$ are sampled from a learned distribution. Conditioned on these variables the continuous variables $\mathbf{x}$ (Surface points and normals) are generated. (b) Latent variables and existences are mapped deterministically to intermediate part representation variables $\mathbf{u}_p$ for each part p. The part representations are then mapped separately to generate data variables. (c) Visible continuous variables are mapped to latent variables through intermediate part representation variables. Learned biases $\mathbf{b}_p$ are introduced for parts p that are not present. (d) The ShapeFA baseline model has linear connections between latent variables, part representations and continuous data variables.

functional relationships, local continuity and smoothness, and be able to handle multi-modality in the data. Consider for example the chair object class: plausible shapes feature chair legs that match in style, and shapes, and attach to the seat appropriately. Chairs can vary at a part level in terms of the style of the chair back or legs, but also at the object level, in which objects can belong to one of a number of distinct styles. Theses styles induce multi-modality in the data-distribution, indeed Huang *et al.* demonstrated that on this dataset the marginal distributions of surface point locations can have complex multi-modal distributions [HKM15].

Although the modelling task is challenging, there are a number of simplifying features that we can exploit. The data-dimensionality is very high, however the data is highly structured, such that the intrinsic dimensionality of the data is much smaller. Take a chair leg for example, in our data this object part consists of around 100 keypoints, but the locations of these keypoints are highly dependent, such that knowledge of only a few points would enable reasonable estimation of the rest. Moreover, the variability across different examples is restricted: legs tend to vary in their length, width, angle, and the location at which they join the base of the chair. As such chair legs possess far fewer degrees of freedom that the 600 dimensions which describe the keypoints and surface normals. We can also make use of the global structure of 3D objects; that they have parts, that the parts are organized in typical arrangements, and so we can design our model in a way that makes use of these known structures. By incorporating a hierarchical part structure in our model, we introduce an inductive bias that favours part decomposability, and a factored relationship between global and local variability. We also can take advantage of recently developed deep learning methods, that can compress the data more effectively than comparable linear methods.

Data description. We represent objects using the following features:

- **Part existences:** For an object class with P possible parts we describe the existence or non-existence of object parts using a binary vector $\mathbf{e} \in \{0,1\}^P$, where part p is present if $e_p = 1$ and not present if $e_p = 0$.
- **Surface points:** The surface of an object can be approximated with a collection of surface points. For some objects not all parts will be present, and so the associated surface points will also not be present. We denote these with a missing data symbol m. For an object with D possible surface points we have a vector $\mathbf{k} \in (\mathbb{R} \cup m)^{3D}$ of point positions. These surface points have a consistent order across examples in a dataset, such that the point dimension k_i^r on object r is in correspondence with point dimension k_i^s on object s.
- **Surface normals:** We can describe the surface of an object in more detail by also including the orientation of the surface points. As with the surface points we describe the surface normals with a vector $\mathbf{n} \in (\mathbb{R} \cup m)^{3D}$.

As our modelling process is identical for keypoints and normals, for convenience we use $\mathbf{x} = [\mathbf{k}, \mathbf{n}]$ to refer to the collection of all continuous variables. Our generative model aims to model the joint distribution of part existences, surface points, and surface normals $p(\mathbf{e}, \mathbf{x})$.

Model structure. We model the joint distribution of existences, points, and surface normals by first modelling the marginal distribution of the existences $p(\mathbf{e})$, and then by modelling the continuous variables conditional on the existences $p(\mathbf{x}|\mathbf{e})$. Let $p(i)$ be the part index associated with keypoint i, and let $\mathcal{M}(\mathbf{e}) = \{i | e_{p(i)} = 0\}$ be the set of indices of missing variables for a particular object. We can then write the set of missing keypoints and normals as $\mathbf{x}_m = \{\mathbf{x}_i = [k_i^1, k_i^2, k_3^3, n_i^1, n_i^2, n_i^3]\}_{i \in \mathcal{M}}$ and the set of visible keypoints as $\mathbf{x}_v = \{\mathbf{x}_i\}_{i \notin \mathcal{M}}$ where we drop the dependence on $\mathbf{e}$ for notational convenience. The missing keypoints and normals $\mathbf{x}_m$ are completely determined by the part existences, and so they are simply assigned the missing data symbol m. Visible keypoints and normals $\mathbf{x}_v$ are generated using a latent variable model. This latent variable model first draws samples $\mathbf{z}$ from a prior Gaussian distribution over a latent space. These prior samples are then mapped to the parameters of a diagonal Gaussian distribution using parameterised functions $\boldsymbol{\mu}(\mathbf{z},\mathbf{e})$ and $\boldsymbol{\sigma}^2(\mathbf{z},\mathbf{e})$ where these parameter functions are given by the decoder of the ShapeVAE:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0},\mathbf{I}) \tag{1}$$

$$p(\mathbf{x}_v|\mathbf{e},\mathbf{z},\boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}_v|\boldsymbol{\mu}(\mathbf{z},\mathbf{e}),\boldsymbol{\sigma}^2(\mathbf{z},\mathbf{e})) \tag{2}$$

$$p(\mathbf{x}_m|\mathbf{e}) = \mathbb{I}[\mathbf{x}_m = [m,\dots,m]]. \tag{3}$$

Figure 2a shows a graphical model representing the model structure. In the following sections we describe the marginal distribution of the part existences, the encoder-decoder structure of the ShapeVAE, the procedure for training the model parameters and the baseline models that we use to evaluate the performance of the ShapeVAE. The following sections assume some knowledge of standard terminology in deep learning, and familiarity with the standard VAE is useful. Full coverage is beyond the scope of this work, and so we refer to Doersch's tutorial for useful background [Doe16].

Part existences. Part existence variables $\mathbf{e}$ are assumed to have been generated by a categorical distribution $p(\mathbf{e}|\phi)$. For an object with P parts this is a distribution over 2^P possible states, which in the absence of independence assumptions requires $2^P - 1$ parameters. For each possible arrangement of part existences $\mathbf{e}$, the maximum likelihood estimate is simply the ration of the number of times the arrangement occurs in the training set $N_\mathbf{e}$ divided by the number of examples in the training set N:

$$p(\mathbf{e}) = \frac{N_\mathbf{e}}{N}, \tag{4}$$

In practice only a few arrangements of part existences occur in object datasets, and so a maximum likelihood estimate assigns most part combinations zero probability. The main limitation of this model is that part arrangements that do not appear in the training set will never be sampled by the generative model. However it is straightforward to choose a prior distribution over the distribution parameters such as a Dirichlet or to even manually choose a desired distribution over part arrangements such that unseen arrangements can be sampled.

ShapeVAE Decoder. The decoder of a variational autoencoder is responsible for mapping from latent variables $\mathbf{z}$ to the parameters $\boldsymbol{\theta}$ of a conditional data distribution $p(\mathbf{x}|\boldsymbol{\theta}(\mathbf{z}))$. This mapping is implemented using a fully-connected neural network. This conditional data distribution is typically chosen such that the data variables $\mathbf{x}$ are conditionally independent given the latent variables $\mathbf{z}$, and so that it naturally models the domain of the data variables. The conditional independence of the data variables forces the latent variables to explain the interdependence of the visible data variables. This causes the model to learn a latent space in which the main modes of variability are captured. In our case we additionally condition on existence variables $\mathbf{e}$, and as the keypoints and surface normal data is continuous we map to the parameters of a diagonal Gaussian distribution $\boldsymbol{\mu}(\mathbf{z},\mathbf{e})$ and $\boldsymbol{\sigma}^2(\mathbf{z},\mathbf{e})$. It is useful to set a minimum variance $\boldsymbol{\sigma}^2_{\text{min}}$ for each dimension of the decoder distribution, so that the total variance is given by $\boldsymbol{\sigma}^2_{\text{tot}} = \boldsymbol{\sigma}^2(\mathbf{z},\mathbf{e}) + \boldsymbol{\sigma}^2_{\text{min}}$. This prevents the decoder from assigning very small variance to any reconstructed training example, which helps reduce overfitting. We treat the minimum variance as a hyperparameter that can be adjusted depending on the task.

We take advantage of the part-structure of the data and use a hierarchical decoder architecture in which global latent variables $\mathbf{z}$ and existence variables $\mathbf{e}$ capture structure, style and shape characteristics of the whole object, and a lower level part-representation $\mathbf{u}_p$ captures variability within each part p. The decoder takes latent variables and existence variables as input, and passes them separately through fully-connected layers of size 256, before concatenating to form a layer of size 512. This intermediate layer is then mapped to the part representation $\mathbf{u}(\mathbf{z},\mathbf{e})$ of size $\sum_k n_{u_p}$ where n_{u_p} is the size of part p's representation. We treat the size of each u_p as a hyperparameter, and typically use 128 or 256 units per part. This representation is then split into its constituent parts $\mathbf{u} = [\mathbf{u}_1,\dots,\mathbf{u}_P]$ and mapped to a fully-connected pre-parameter layer $\mathbf{h}_p(\mathbf{z},\mathbf{e})$ of size 512. Finally the output parameters are obtained using a linear layer for the mean $\boldsymbol{\mu}_p(\mathbf{z},\mathbf{e}) = \texttt{Linear}(\mathbf{h}_p(\mathbf{z},\mathbf{e}))$, and by applying a soft-plus non-linearity for the variance $\boldsymbol{\sigma}^2_p(\mathbf{z},\mathbf{e}) = \texttt{softplus}(\texttt{Linear}(\mathbf{h}_p(\mathbf{z},\mathbf{e})))$ to ensure that the variance is positive. A simplified view of the decoder structure is shown in Figure 2b.

ShapeVAE Encoder. The encoder of the ShapeVAE aims to approximate the posterior distribution over the latent variables $p(\mathbf{z}|\mathbf{x},\mathbf{e})$ associated with the generator distribution $p(\mathbf{x}|\mathbf{z},\mathbf{e})$. As the true posterior distribution $p(\mathbf{z}|\mathbf{x},\mathbf{e})$ is intractable to evaluate, we make use of an approximate posterior $q(\mathbf{z}|\mathbf{x},\mathbf{e})$, and use variational inference to learn the parameters of both the encoder and decoder simultaneously. Similar to the decoder, the ShapeVAE decoder is a neural network that maps from inputs $\mathbf{x}$ to the parameters of a diagonal Gaussian $\boldsymbol{\mu}(\mathbf{x})$ and $\boldsymbol{\sigma}^2(\mathbf{x})$.

The ShapeVAE encoder reverses the architecture of the decoder but is modified in order to process input parts which may be missing. As shown in Figure 2c the encoder takes keypoints and normals $\mathbf{x}$ as input and maps to a part representation $\mathbf{u} = [\mathbf{u}_1,\dots,\mathbf{u}_P]$. For parts that are present, the input is mapped through an intermediate fully connected layer of size 512, whereas parts that are missing simply generate a learnable bias $\mathbf{b}_p$ which is added in the appropriate position to the the part representation. The part representation is then concatenated and passed through to a fully-connected pre-parameter layer of size 512. As with the decoder the pre-parameter layer is mapped through linear and soft-plus layers to obtain means and diagonal variances of the encoder distribution.

Training. We learn the parameters of the ShapeVAE encoder $\boldsymbol{\theta}$ and decoder $\boldsymbol{\phi}$ using the auto-encoding variational Bayes algorithm [KW14]. This algorithm maximizes a variational lower-bound $\mathcal{L}(\boldsymbol{\theta},\boldsymbol{\phi},\mathbf{x})$ on the true log-likelihood:

$$\mathcal{L}(\boldsymbol{\theta},\boldsymbol{\phi},\mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] - \alpha D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})). \tag{5}$$

In the general case, the lower-bound is intractable, and so it is instead approximated using a Monte-Carlo estimator, and optimized using gradient based methods. For further details see [KW14].

The first term in the lower bound is the expected reconstruction probability with respect to the encoder distribution. This term encourages the decoder to reconstruct its input from samples drawn from the encoder distribution. The second term is the KL divergence between the the ShapeVAE encoder distribution, and the prior distribution over the latent space $p(\mathbf{z})$. The KL divergence is an asymmetric measure of distance between two probability distributions, and maximizing the negative KL divergence encourages the latent posterior to be close to the prior. In practice this has the effect of pushing the means of the encoder distribution towards 0, and the variances towards 1. This works as a regularizer, as the decoder must be robust to the noisy inputs provided by samples from the encoder distribution. The alpha parameter in the variational lower bound allows the KL term to be weighted more or less strongly in the optimization.

Baseline models. In order to evaluate the ShapeVAE we introduce two baseline models. The first is a very simple baseline, in which a separate diagonal Gaussian model is fitted to the points and surface normals for each combination of existences in the training set. The model is given by:

$$p(\mathbf{x}_v|\mathbf{e},\boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}_v|\boldsymbol{\mu}_\mathbf{e},\boldsymbol{\sigma}^2_\mathbf{e}). \tag{6}$$

Thus for each pattern of existences in the training set we take the mean of the surface points and normals, and compute the variance of each dimension.

We also introduce a more competitive baseline, the shape factor analyzer (ShapeFA). In this model we replicate the hierarchical structure of the ShapeVAE, but use linear connections between layers, rather than arbitrary non-linear functions. This is inspired by the widely-used factor analysis model in which latent variables are linearly mapped to data variables, and diagonal Gaussian noise is added. In our version, the model has two layers of linear mappings: from top-level structural latent variables to a part representation, and from the part representation to the object parts. As in the ShapeVAE the top-level latent variables capture overall shape variability in terms of style, symmetry and functional dependencies, while the lower level latent variables capture local variability within a particular part. Writing $\mathbf{u}_v$ for the set of visible part representation variables we have the following model:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0},\mathbf{I}) \tag{7}$$

$$p(\mathbf{u}_v|\mathbf{z},\mathbf{e},\boldsymbol{\theta}) = \mathcal{N}(\mathbf{u}_v|\mathbf{W}^{(\mathrm{T})}_\mathbf{e}\mathbf{z}+\boldsymbol{\mu}^{(\mathrm{T})}_\mathbf{e},\boldsymbol{\Psi}^{(\mathrm{T})}_\mathbf{e}) \tag{8}$$

$$p(\mathbf{x}_p,\mathbf{n}_p|\mathbf{u}_p,\boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}_p|\mathbf{W}^{(\mathrm{B})}_p\mathbf{u}_p+\boldsymbol{\mu}^{(\mathrm{B})}_p,\boldsymbol{\Psi}^{(\mathrm{B})}_p), \tag{9}$$

Where we use $\boldsymbol{\theta}^{(\mathrm{B})}$ and $\boldsymbol{\theta}^{(\mathrm{T})}$ to denote bottom and top-layer parameters respectively. The advantage of this model is that by integrating out the part representation variables we can evaluate the log-likelihood exactly. We can also train it rapidly using the greedy layer-wise procedure described by Tang *et al.* [TSH12]. Figure 2d shows the structure of the ShapeFA.

5. Evaluation

We detail the performance of the ShapeVAE on shape completion, and test set log-likelihood tasks, as well demonstrate the model's ability to synthesize 3D shapes. We examine features of the latent-space learned by the ShapeVAE and compare surface reconstruction methods that can be used to convert sampled point clouds to

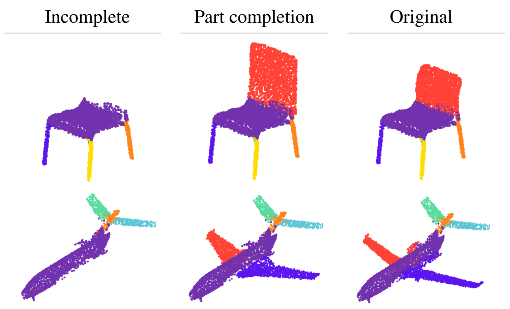


Figure 3: **Shape completion**. (Incomplete) Unseen examples with missing parts. (Part completion) The missing parts are completed by conditional inference with the ShapeVAE. (Original) Original object.

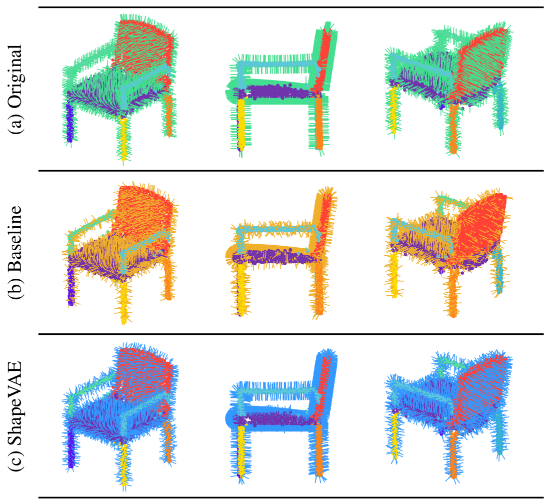


Figure 4: **Point orientation recovery.** (a) A test-set example shown with ground-truth point orientations. (b) Point orientations estimated using local plane fitting [HDD*92]. (c) Point orientations predicted by performing conditional-inference with the ShapeVAE.

	Shape completion		
Model	Chairs	Planes	Bikes
ShapeVAE-8	0.124	0.106	**0.168**
ShapeVAE-64	0.121	**0.104**	**0.168**
ShapeFA-8	**0.116**	0.243	0.178
ShapeFA-64	0.155	0.267	0.255

	Log-likelihood		
Model	Chairs	Planes	Bikes
ShapeVAE-8*	0.454	0.466	0.158
ShapeVAE-64*	0.486	0.551	0.206
ShapeFA-8	**0.514**	**0.739**	**0.475**
ShapeFA-64	0.466	0.648	0.300
Diag. Gaussian	0.001	0.057	−0.015

meshes. For all plots of object samples or reconstructions we show the mean $\mathbb{E}[\mathbf{x}]_{p(\mathbf{x}|\mathbf{z})}$ of the data variables given the latents.

Training details. We trained ShapeVAEs with 8 and 64 latent dimensions and part representations of size 256 per part using the ADAM optimizer [KB15] with a learning rate of 10^{-4} for up to 500 passes through the training set in all our experiments. For the quantitative experiments we used early stopping, where model training was halted based on performance a validation set. For sample synthesis we allowed the model to train for the full 500 passes through the training set. For the airplane and chair datasets we used a batch size of 100, and for the bike dataset we use a batch size of 64. We use a minimum variance of $\sigma^2_{\text{min}} = 10^{-3}$ and KL weighting $\alpha = 10^2$ in all our models. ShapeFAs with 8 and 64 latent dimensions, and part representations of size 128 per part were trained as a baseline model. The ShapeVAE takes around 30 minutes to train on an Nvidia Titan X GPU for our largest dataset consisting of 3701 chair. This is a order of magnitude faster than the beta shape machine introduced by Huang *et al.* which is reported to take 45 hours for the same dataset [HKM15].

Shape completion. We evaluate the ShapeVAE's ability to complete shapes, where the task is to infer missing keypoints or surface normal variables given observations of the other variables. For the ShapeVAE we perform conditional inference by initializing the missing values with noise, and iteratively sampling the latent variables conditioned on the data, and then the data variables given the latents. This defines an MCMC chain that samples from the conditional distribution as required [RMW14]. Two particularly useful tasks are the completion of missing parts, and estimation of surface normals for a given point cloud. Figure 3 shows examples of plausible completions for the part-completion task, and Figure 4 shows normals recovered by the ShapeVAE given an input point cloud.

In order to quantitatively test the shape completion abilities of the ShapeVAE, we use the following experiment. We sample with replacement 1000 objects from the test sets of each object class. We then drop-out parts independently with probability 0.25, and reject a part selection if all or none of the parts remain. The ShapeVAE and ShapeFA are used to impute the missing parts and we compute the mean squared error between the reconstructed values and the true values. For both the ShapeVAE and ShapeFA we sample from the conditional distribution of the missing parts given the visible parts, and estimate the conditional mean by averaging over 25 samples, with an MCMC burn-in of 100 samples, and a gap of 10 between each chosen sample. This conditional mean is used as the reconstruction estimate. The ShapeVAE outperforms the ShapeFA in both the planes and bikes categories, and achieves similar results in the chairs category as shown in Table **??**.

Likelihood. A standard measure of a generative model's performance is test set log-likelihood: the average probability of a set of unseen datapoints under the model. We evaluate the log-likelihood obtained by the ShapeVAE against baseline models on each of the four object classes. Although the ShapeVAE is a probabilistic model it is not possible to evaluate the exact probability of a data point, only a lower bound. As such we estimate the log-likelihood using 10,000 importance-weighted samples [BGS15]. This provides a lower bound on the log-likelihood that is tighter

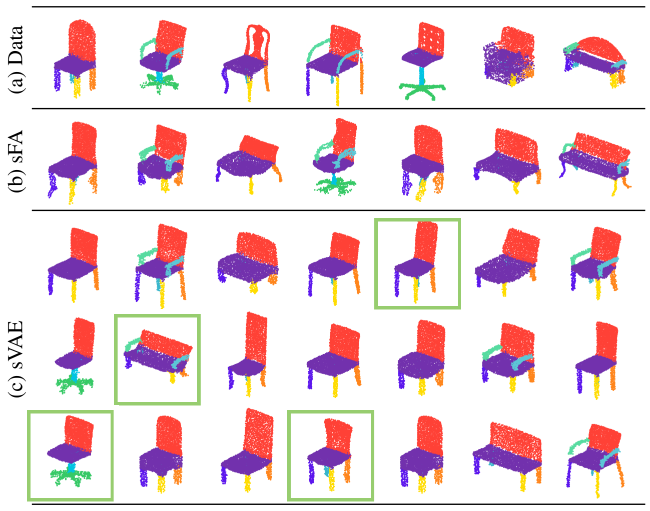


Figure 5: **Shape samples**. (a) A collection of examples from the chairs dataset. (b) Samples generated by a ShapeFA with 64 latent dimensions. (c) Samples generated by a ShapeVAE with 64 latent dimensions. Stylistic variability highlighted in green.

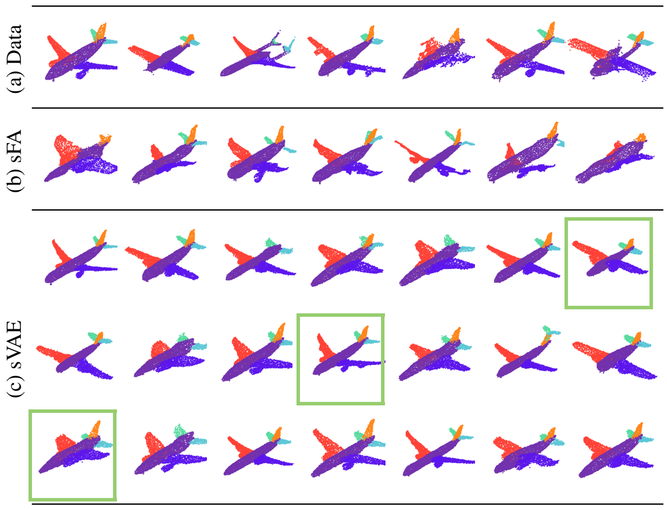


Figure 6: **Shape samples**. (a) A collection of examples from the airplanes dataset. (b) Samples generated by a ShapeFA with 64 latent dimensions. (c) Samples generated by a ShapeVAE with 64 latent dimensions. Stylistic variability highlighted in green.

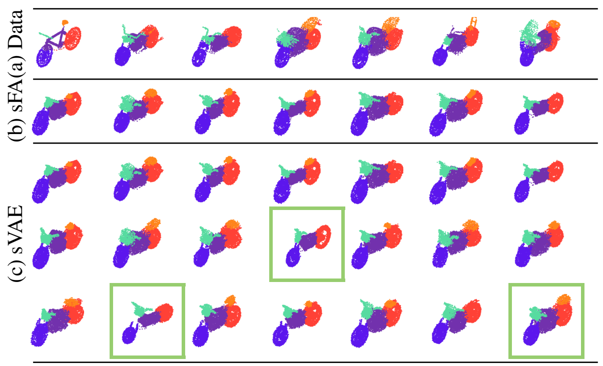


Figure 7: **Shape samples**. (a) A collection of examples from the bike dataset. (b) Samples generated by a ShapeFA with 64 latent dimensions. (c) Samples generated by a ShapeVAE with 64 latent dimensions. Shape variability highlighted in green.

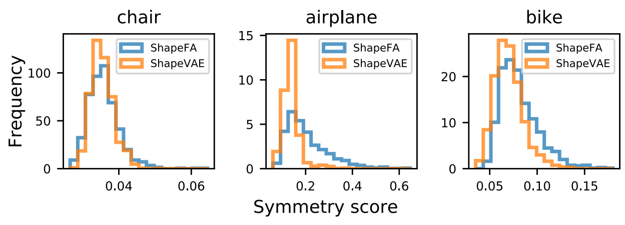


Figure 8: **Symmetry.** Symmetry score distribution for samples from the ShapeVAE and the ShapeFA.

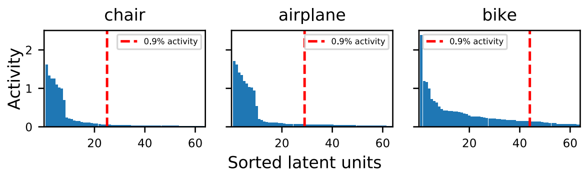


Figure 9: **Latent space activity.** The activity of each latent unit (as defined in Section 5: Latent structure) sorted in descending order for ShapeVAEs with 64 latent dimensions. The first units at which the cumulative activity is greater than 90% is indicated.

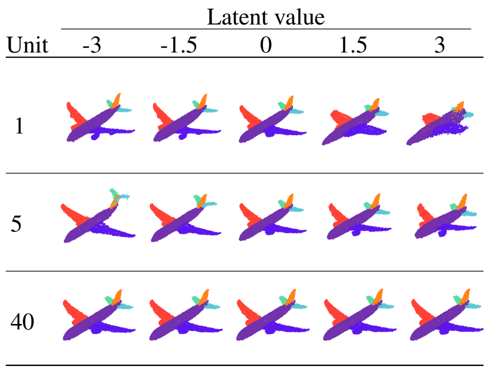


Figure 10: **Latent features.** Data reconstructions obtained by varying a single latent unit through $[-3,\ldots,3]$ while setting the others to zero. Units are numbered in order of their activity (see Figure 9.

than the training lower bound, however it may still be significantly lower than the true value. We compare with the ShapeFA and diagonal Guassian baselines for which we can obtain an exact log-likelihood score. Table **??** shows that although the ShapeVAE has much better performance than the Gaussian baseline, the ShapeFA with 8 latent dimensions achieves the best performance on all datasets. However it should be emphasized that the reported score for ShapeVAE models is a lower bound, and the true log-likelihood score may be significantly higher.

Sample quality. We qualitatively investigate the extent to which the ShapeVAE synthesizes quality 3D objects, as compared with real data examples and the ShapeFA. Good samples are characterised by realism: where objects demonstrate regular surfaces, appropriate symmetry, functional plausibility in terms of part attachments, as well as fine detail. Samples should also demonstrate a wide range of shape variability, and capture the main modes of variation in the object class. For example a model trained on chairs data should be able to generate both wide benches, tall, thin chairs, as well as armchairs and office chairs. Another desirable feature of a model's samples is novelty: shape samples should not simply recreate instances from the training data set.

Figures 5, 6 and 7 show samples from the ShapeVAE alongside data examples, and samples from the ShapeFA. The ShapeFA produces samples that demonstrate a wide range of variability, however they feature irregular surfaces and occasional asymmetry. This is particularly evident for the airplane object class in which a number of examples demonstrate misshapen features. By contrast the ShapeVAE produces samples that are realistic, with regular surfaces and good symmetry. The samples also demonstrate a good range of shape variability in terms of object style, with office chairs, benches, tall chairs and standard four-leg chairs all well represented for the chair object class, commercial jets, fighter jets and small planes present in the airplane samples, and bulky motorbikes and smaller off-road bikes present in the bike samples. However, neither the ShapeVAE or ShapeFA produce samples with the level of fine-detail present in the data examples. Features like chair backs with slats, or ornamental legs are not present in the model samples. This is consistent with the blurry image samples produces using VAEs in the machine learning literature [LDC16]. It is arguable that the assumption of independence of the data variables given the latent variables enables the VAE to simply model fine details as noise.

To assess the extent to which the ShapeVAE produces symmetrical samples, we compute a symmetry score. The symmetry score for a particular object is defined as the average euclidean distance between sampled points, and their nearest neighbors, after flipping on some axis of symmetry. For chairs we use the axis that splits the seat and back in half, for airplanes, we flip on the axis that extends lengthways through the fuselage. Figure 8 shows the distribution of symmetry scores for 1000 samples from a ShapeVAE with 64 latent dimensions, and a ShapeFA with 64 latent dimensions. The ShapeVAE achieves better symmetry scores, with a notable difference for the airplanes dataset.

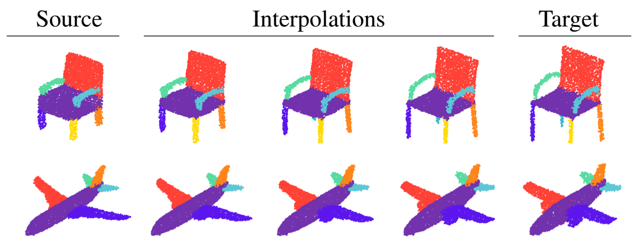


Figure 11: **Latent space interpolation.** (Source) Source 3D point cloud. (Target) Target 3D point cloud. (Interpolations) ShapeVAE Interpolations obtained by a linear interpolation in latent space.

Latent structure. In order to examine the latent structure learned by the ShapeVAE we train a model with 2 latent dimensions, and plot the latent space along with embeddings of a selection of the training set in Figure 1 (left). The training set embeddings are obtained by passing the input examples through the ShapeVAE encoder. We see that semantically similar chair styles such as benches, arm-chairs, tall chairs and ordinary chairs form clusters in latent space, which indicates that the ShapeVAE with 2 latent dimesions has learned global geometric structure such as the relative height, width and depth of the 3D objects. By selecting points close to these clusters, we can decode and obtain samples of 3D objects (Figure 1 (middle)). The samples share some of the main semantic characteristics of the training examples that they are embedded close to. In this case the relatively narrow range of variability of the sampled shapes can be attributed partly to the very low-dimensional latent space.

For models with more latent dimensions it isn't possible to visualise the latent space in the same way, however we can demonstrate some qualities of higher dimensional latent spaces by tracing a straight line in latent space, and visualising the decoded objects sampled along this line. Figure 11 shows examples of these interpolations achieved using ShapeVAEs with 64 latent dimensions. The interpolations are smooth, and each intermediate point produces a plausible 3D object.

In previous work on VAEs it has been demonstrated that the models often do not encode much information in a number of the latent variables [BGS16, SRM*16]. We use the measure of activity described in [BGS16] which is the the variance across the data set of examples transformed using the mean of the encoder distribution for a particular dimension. If a latent unit varies across different data examples, then it is reasonable to think that it is encoding information useful for reconstruction. We plot the activity of the latent units for ShapeVAEs with 64 latent dimensions in Figure 9. The figure shows that for each object class, the activity of the latent units drops significantly after about 10 units. This indicates that the effective dimensionality of the ShapeVAE can be lower than the pre-specified number of latent units. In Figure 10 we demonstrate some examples of features learned by different latent dimensions by varying a particular unit while keeping the others fixed. We see that features encoded by latent units with high activity (unit 1, 5) encode more significant shape changes than those with low activity (unit 40). This reinforces the notion that the latent activity metric captures the importance of the latent units.

Surface reconstruction. We reconstruct 3D meshes from sampled point clouds using three methods: alpha shapes [EKS83], template deformation [SSP07] and Poisson surface reconstruction [KBH06]. For alpha shapes we use a radius $r = 0.12$. For template deformation we use a variant of an embedded deformation in which the deformation graph's nodes are the keypoints of template part, and we smoothly deform so as to match the corresponding samples' keypoints. For Poisson surface reconstruction we the implementation of Kazhdan *et al.* with default parameters except for the number of samples per node, which we set to 1.5 [KBH06].

Exemplar mesh reconstructions using alpha shapes, template deformation and Poisson surface reconstruction for surface point clouds sampled from trained ShapeVAEs are shown in Figure 12. We qualitatively evaluate the mesh reconstructions in terms of mesh quality and faithfulness of the reconstructed surface to the shape of the sampled points. The alpha shapes reconstructions (Figure 12b) are successful in capturing the coarse shape of the sampled points, however as the method relies on reconstruction of the input points, it produces noisy and uneven output. Triangulation-based methods also suffer at object part boundaries where surfaces intersect. This is clearly illustrated by the wing-fuselage intersection on the airplane example. The template deformation reconstructions (Figure 12c) make use of a database of existing object parts, which are then deformed and repositioned so as to match the sampled points. As such they inherit good-quality, human-designed mesh parts, which combine to create a high-quality, noise-free mesh reconstruction. However, template meshes can only be deformed to a limited extent before becoming irregular, and so the extent to which a template deformation can match an object sample is limited. In addition if segmentations of meshes in the database are not exactly aligned with real part boundaries, then issues can arise at the intersection of parts in the synthesized mesh. This is highlighted in the chairs example in which the chair arms are not quite compatible with the chair seat. Figure 12d shows Poisson surface reconstructions using the sampled point normals. The reconstructions are of better quality than alpha shapes with respect to noise and artifacts, and faithfully recover the shape of the point samples. However it should be noted that in cases where the sampled points are sparse, the Poisson surface reconstruction can fail, resulting in unusable reconstructions.

6. Discussion

In this paper we have demonstrated that the ShapeVAE can effectively model the high dimensional distribution over object shapes, producing realistic and novel samples. We show that modern deep learning methods are effective at scaling to very high dimensionality data, and that by modelling both 3D surface points and surface normals we enable the use of normal-based surface reconstruction methods. Furthermore our models can be trained very quickly in comparison to other methods, which increases the accessibility of these methods to practitioners.

The most significant limitation of our method is the reliance on input datasets containing consistent mesh segmentations as well as dense correspondences. Although there exist methods for the automatic establishment of correspondences and segmentations, these methods are imperfect and can result in poor outputs. A

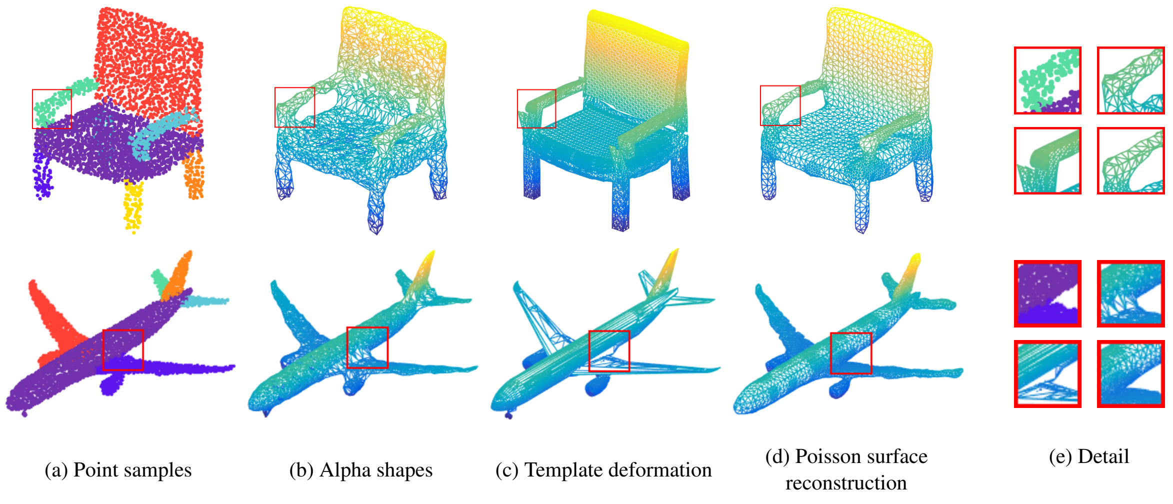


Figure 12: **Mesh reconstruction**. (a) A sampled 3D point cloud with surface normals. Surface reconstruction using (b) alpha shapes, (c) template deformation, and (d) Poisson surface reconstruction. (e) Surface reconstruction detail for (top left) point samples, (top right) alpha shapes, (bottom left) template deformation and (bottom right) Poisson surface reconstruction.

significant issue is is that the notion of one-to-one point correspondences for objects in diverse datasets such as chairs is ill-founded. The result is that the input data for our method can feature poor correspondences, which has a knock-on effect on sample quality. We believe that a promising avenue for future research is to represent objects using unordered point sets, which would enable the use of large datasets without pre-processing, and correspondence quality issues. Some work has already taken place in this area, with deep learning methods applied to 3D point sets for the purpose of object classification, semantic scene parsing and part segmentation [QSMG16]. We believe there is potential to modify these methods for generative modelling, which would enable the synthesis of arbitrary point clouds.

Although the ShapeVAE's samples display a good range of variability, they are somewhat lacking in fine detail in comparison with the input point sets. This is an issue that has been documented in the machine learning literature, in which VAE-based generative models of images demonstrate blurriness and a lack of detail [DB16]. This effect has been attributed to the use of unimodal generative distributions such as Gaussians. One solution to this issue that has emerged in generative models is the use of generative adversarial networks (GANs), which demonstrate mode-seeking behaviour, and thus produce samples closer to the true data manifold than alternative approaches. As such, a potential future direction is to make use of GAN with a similar architecture to the ShapeVAE decoder to generate objects.

7. Acknowledgements

The work of CW is supported in part by EPSRC grant EP/N510129/1. CN is supported by the Centre for Doctoral Training in Data Science, funded by EPSRC grant EP/L016427/1. We thank Haibin Huang, Vangelis Kalogerakis and Ben Marlin for sharing their data and providing information about their methods, and Niloy Mitra for his valuable comments on an early draft.

References

[AKZM14] AVERKIOU M., KIM V. G., ZHENG Y., MITRA N. J.: Shapesynth: Parameterizing model collections for coupled shape exploration and synthesis. *Comput. Graph. Forum 33*, 2 (2014), 125–134. 2

[BGS15] BURDA Y., GROSSE R. B., SALAKHUTDINOV R.: Importance weighted autoencoders. *CoRR abs/1509.00519* (2015). 7

[BGS16] BURDA Y., GROSSE R., SALAKHUTDINOV R.: Importance weighted autoencoders. 9

[BLRW16] BROCK A., LIM T., RITCHIE J. M., WESTON N.: Generative and discriminative voxel modeling with convolutional neural networks. *CoRR abs/1608.04236* (2016). 3

[CSCS15] CHOY C. B., STARK M., CORBETT-DAVIES S., SAVARESE S.: Enriching object detection with 2D-3D registration and continuous viewpoint estimation. In *CVPR* (2015), IEEE Computer Society, pp. 2512–2520. 2

[CTCG95] COOTES T. F., TAYLOR C. J., COOPER D. H., GRAHAM J.: Active shape models-their training and application. *Computer Vision and Image Understanding 61*, 1 (1995), 38–59. 2

[DB16] DOSOVITSKIY A., BROX T.: Generating images with perceptual similarity metrics based on deep networks. In *NIPS* (2016), pp. 658–666. 10

[DCSF15] DENTON E. L., CHINTALA S., SZLAM A., FERGUS R.: Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS* (2015), pp. 1486–1494. 3

[Doe16] DOERSCH C.: Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908* (2016). 5

[EHWW14] ESLAMI S. M. A., HEESS N., WILLIAMS C. K. I., WINN J. M.: The shape boltzmann machine: A strong model of object shape. *International Journal of Computer Vision 107*, 2 (2014), 155–176. 3

[EKS83] EDELSBRUNNER H., KIRKPATRICK D. G., SEIDEL R.: On the shape of a set of points in the plane. *IEEE Trans. Information Theory 29*, 4 (1983), 551–558. 9

[FAvK*14] FISH N., AVERKIOU M., VAN KAICK O., SORKINE-HORNUNG O., COHEN-OR D., MITRA N. J.: Meta-representation of shape families. *ACM Trans. Graph. 33*, 4 (2014), 34:1–34:11. 2

[FSG16] FAN H., SU H., GUIBAS L.: A point set generation network for 3d object reconstruction from a single image. *arXiv preprint arXiv:1612.00603* (2016). 3

[GFRG16] Girdhar R., Fouhey D. F., Rodriguez M., Gupta A.: Learning a predictable and generative vector representation for objects. In *ECCV* (2016). 2

[GPM∗14] Goodfellow I. J., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A. C., Bengio Y.: Generative adversarial nets. In *NIPS* (2014), pp. 2672–2680. 3

[HDD∗92] Hoppe H., DeRose T., Duchamp T., McDonald J. A., Stuetzle W.: Surface reconstruction from unorganized points. In *SIGGRAPH* (1992), ACM, pp. 71–78. 6

[HKM15] Huang H., Kalogerakis E., Marlin B. M.: Analysis and synthesis of 3D shape families via deep-learned generative models of surfaces. *Computer Graphics Forum 34*, 5 (2015), 25–38. 2, 3, 4, 7

[HM09] Heimann T., Meinzer H.: Statistical shape models for 3d medical image segmentation: A review. *Medical Image Analysis 13*, 4 (2009), 543–563. 2

[KB15] Kingma D., Ba J.: Adam: A method for stochastic optimization. *Proceedings of the International Conference on Learning Representations* (2015). 7

[KBH06] Kazhdan M. M., Bolitho M., Hoppe H.: Poisson surface reconstruction. In *Symposium on Geometry Processing* (2006), vol. 256 of *ACM International Conference Proceeding Series*, Eurographics Association, pp. 61–70. 9

[KCKK12] Kalogerakis E., Chaudhuri S., Koller D., Koltun V.: A probabilistic model for component-based shape synthesis. *ACM Trans. Graph. 31*, 4 (2012), 55:1–55:11. 2

[KLM∗13] Kim V. G., Li W., Mitra N. J., Chaudhuri S., DiVerdi S., Funkhouser T. A.: Learning part-based templates from large collections of 3D shapes. *ACM Trans. Graph. 32*, 4 (2013), 70:1–70:12. 3

[KW14] Kingma D. P., Welling M.: Auto-encoding variational Bayes. In *Proceedings of the International Conference on Learning Representations* (2014). 3, 5

[LDC16] Lamb A., Dumoulin V., Courville A.: Discriminative regularization for generative models. *arXiv preprint arXiv:1602.03220* (2016). 8

[LS10] Liebelt J., Schmid C.: Multi-view object class detection with a 3D geometric model. In *CVPR* (2010), IEEE Computer Society, pp. 1688–1695. 2

[LTC97] Lanitis A., Taylor C. J., Cootes T. F.: Automatic interpretation and coding of face images using flexible models. *IEEE Trans. Pattern Anal. Mach. Intell. 19*, 7 (1997), 743–756. 2

[MDH12] Mohamed A., Dahl G. E., Hinton G. E.: Acoustic modeling using deep belief networks. *IEEE Trans. Audio, Speech & Language Processing 20*, 1 (2012), 14–22. 3

[PT11] Payet N., Todorovic S.: From contours to 3D object detection and pose estimation. In *ICCV* (2011), IEEE Computer Society, pp. 983–990. 2

[QSMG16] Qi C. R., Su H., Mo K., Guibas L. J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. *CoRR abs/1612.00593* (2016). 2, 10

[RHSW11] Roux N. L., Heess N., Shotton J., Winn J. M.: Learning a generative model of images by factoring appearance and shape. *Neural Computation 23*, 3 (2011), 593–650. 3

[RMW14] Rezende D. J., Mohamed S., Wierstra D.: Stochastic backpropagation and approximate inference in deep generative models. In *ICML* (2014), vol. 32 of *JMLR Workshop and Conference Proceedings*, JMLR.org, pp. 1278–1286. 7

[SH09] Salakhutdinov R., Hinton G. E.: Deep boltzmann machines. In *AISTATS* (2009), vol. 5 of *JMLR Proceedings*, JMLR.org, pp. 448–455. 3

[SHM∗14] Su H., Huang Q., Mitra N. J., Li Y., Guibas L. J.: Estimating image depth using shape collections. *ACM Trans. Graph. 33*, 4 (2014), 37:1–37:11. 2

[SRM∗16] Sønderby C. K., Raiko T., Maaløe L., Sønderby S. K., Winther O.: Ladder variational autoencoders. In *Advances in Neural Information Processing Systems* (2016), pp. 3738–3746. 9

[SSP07] Sumner R. W., Schmid J., Pauly M.: Embedded deformation for shape manipulation. *ACM Trans. Graph. 26*, 3 (2007), 80. 9

[TDB17] Tatarchenko M., Dosovitskiy A., Brox T.: Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. *arXiv preprint arXiv:1703.09438* (2017). 2

[TSH12] Tang Y., Salakhutdinov R., Hinton G. E.: Deep mixtures of factor analysers. In *ICML* (2012), icml.cc / Omnipress. 6

[WSK∗15] Wu Z., Song S., Khosla A., Yu F., Zhang L., Tang X., Xiao J.: 3D shapenets: A deep representation for volumetric shapes. In *CVPR* (2015), IEEE Computer Society, pp. 1912–1920. 2

[WZX∗16] Wu J., Zhang C., Xue T., Freeman B., Tenenbaum J.: Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *NIPS* (2016), pp. 82–90. 2, 3

[YAMK15] Yümer M. E., Asente P., Mech R., Kara L. B.: Procedural modeling using autoencoder networks. In *UIST* (2015), ACM, pp. 109–118. 2

[ZB15] Zuffi S., Black M. J.: The stitched puppet: A graphical model of 3D human shape and pose. In *CVPR* (2015), IEEE Computer Society, pp. 3537–3546. 2

[ZCM13] Zheng Y., Cohen-Or D., Mitra N. J.: *Smart Variations*: Functional substructures for part compatibility. *Comput. Graph. Forum 32*, 2 (2013), 195–204. 2

[ZSSS13] Zia M. Z., Stark M., Schiele B., Schindler K.: Detailed 3D representations for object recognition and modelling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 35*, 11 (2013), 2608–2623. 2