manuscript No.
(will be inserted by the editor)

LPMNet: Latent Part Modification and Generation for 3D Point Clouds

Cihan Öngün · Alptekin Temizel

Abstract In this paper, we focus on latent modification and generation of 3D point cloud object models with respect to their semantic parts. Different to the existing methods which use separate networks for part generation and assembly, we propose a single end-to-end Autoencoder model that can handle generation and modification of both semantic parts, and global shapes. The proposed method supports part exchange between 3D point cloud models and composition by different parts to form new models by directly editing latent representations. This holistic approach does not need part-based training to learn part representations and does not introduce any extra loss besides the standard reconstruction loss. The experiments demonstrate the robustness of the proposed method with different object categories and varying number of points. The method can generate new models by integration of generative models such as GANs and VAEs and can work with unannotated point clouds by integration of a segmentation module.

Keywords Point cloud · Autoencoder · GAN · VAE · Part interpolation

1 Introduction

Deep learning applications in the 3D domain are becoming increasingly more popular, expanding on the already successful applications in the 2D image domain and there is a surge in the number of studies focusing on the artificial generation of 3D models. Artificially generated 3D models have many uses in virtual environments, simulations, and 3D printing. Leading companies are now providing AI tools that help users create better 3D models, make recommendations for more realistic models and correct errors in graphics for a better user experience.

A number of different data types can be used to represent 3D models. While mesh-based representation is popular in computer graphics, voxel-based representation is preferred in 3D data processing applications because of its simplicity. On the other hand, point clouds are the most prominent data type in 3D perception of the real world and they are popular in various fields such as 3D scanners, robotics, autonomous cars, face recognition, and human pose estimation. Detection, recognition and segmentation are the main tasks in these fields and generation of 3D models in point clouds is expected to facilitate new types of approaches for these tasks.

Real-world objects are composed of individual parts and model generation systems should ideally be part-aware in-line with this semantic composition. The basic approach in the literature is to generate parts separately and then assemble them to form the complete object. However, this approach needs training different networks which are experts on specific parts and a separate network to combine these parts. In this paper, we propose a holistic approach to learn the semantic properties of the parts with a single neural network model. The proposed architecture is an Encoder-Decoder network that represents the parts, in addition to the global shape, separately in the feature space. Making modifications in the feature space allows meaningful modifications by preserving semantic properties. This is in contrast to the traditional way of making modifications in the input space which results in a completely new model. The contributions of the proposed method are as follows:

Cihan Öngün · Alptekin Temizel
Graduate School of Informatics, Middle East Technical University, Ankara, Turkey
E-mail: {congun,atemizel}@metu.edu.tr

- It handles part editing, modification and global model generation with a single architecture and eliminates the need for an additional network for part assembly. The parts generated by modifications of latent space stay coherent with the global shape.
- It does not require any additional loss function other than the standard reconstruction loss.
- It provides a generic solution to convert regular generative networks based on PointNet feature extraction into part-aware networks.
- It is scalable and can be used with different point cloud sizes, objects having different numbers of parts and parts having different resolutions.
- It can process models without any explicit part information during inference by integration of a segmentation module.

The paper is structured as follows: Section 2 summarizes the literature on point cloud generation with necessary background information. Section 3 explains the proposed method in detail. Section 4 gives the details of the experiments and the visualization of sample results. Section 5 provides the conclusions and directions for future work.

2 Background and Related Work

2.1 Point clouds

Point clouds are a set of unstructured points in a 3D coordinate system that defines 3D models. Capturing, visualizing and modification of point clouds are simpler compared to the other 3D representation methods since the data points only have position variables for a point p and some extra information such as color value when needed. A 3D model can be defined by a varying number of points and the higher the number points, the better and more detailed is the representation. While capturing and modification of point clouds is straightforward, the processing in this domain is challenging due to the following properties:

Point clouds are unstructured and points have no connectivity information. The nearest or sequential points cannot be assumed to be neighbors since they may be in different semantic parts. The proposed method uses a point-wise feature extractor to process points independently without any connection information.

Points in a point cloud model can be in any order. A point cloud with N points can be defined by $N!$ permutations of ordering. The proposed method uses order invariant part and global feature extractors to deal with the ordering problem.

Point clouds can have arbitrary number of points. The number of points is not constant and can be increased or decreased to have different resolutions. However, most of the models assume a fixed input size. The proposed method utilizes max-pooling operation to extract the important points for feature extraction allowing use of an arbitrary number of points.

PointNet [3] is the most popular neural network based approach for point cloud processing. It provides an end-to-end solution to extract global and local features and it is an effective baseline for a range of tasks such as object classification, part segmentation, and scene semantic parsing. PointNet++ [14] is an extended version of the original PointNet which uses a hierarchical neural network that applies PointNet recursively on a nested partitioning of the input point set. PointNet++ uses sampling and grouping layers to extract features from local point neighborhoods. Neighboring points may belong to different parts, so these layers must also be redesigned for part considerations. As the proposed method introduces a new step for part feature extraction in intermediate layers, it would not be possible to use PointNet++ directly. Hence the standard PointNet is adopted since it provides a holistic approach for feature extraction.

Some approaches convert point clouds into different representations to tackle with the aforementioned problems. DeepSDF [13] uses Signed Distance Functions to represent 3D shapes with continuous functions for easier processing of them in neural networks. While continuous functions do not suffer from the same problems as point clouds, pre-processing and post-processing steps are necessary for conversion. Also, it is not straightforward to represent semantic parts of 3D shapes with continuous functions.

2.2 Generative Models

Generative Adversarial Networks (GAN) [6] consist of 2 different neural networks; Generator G and Discriminator D. While the Generator generates new realistic samples, Discriminator aims to distinguish between real and fake samples and it is trained by a loss measure calculating the difference between the predictions and true values. Generator aims to fool the Discriminator so it needs to generate as realistic samples as possible. At each iteration, Discriminator gets better at distinguishing real and fakes samples and Generator gets better at generating more realistic samples. The whole system is a minimax game between Generator and Discriminator. Assuming x is real data and z is a latent variable, GAN

loss function can be defined as:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[log\ D(x)] + E_{z\sim p_z(z)}[log\ (1-D(G(z)))] \quad (1)$$

Variational Autoencoder (VAE) [8] architecture is an extension of Autoencoder (AE) architecture addressing the content generation problem and the main difference lies in the bottleneck layer. AEs represent each input sample with a latent variable in a lower dimension. This may lead to an overfitting problem since the network is not trained for a regularized latent space. Latent space may not be continuous and some points in this latent space may represent meaningless samples in the input space. VAEs represent each input sample with a distribution by adding a regularization loss to the reconstruction loss. Regularization imposes latent space to belong to a standard normal distribution so any random point generates a new meaningful sample.

A comprehensive analysis of different point cloud generation models is provided in [1] where the PointNet model is used as an Encoder and a multi-layer perceptron is used as a Decoder. Chamfer Distance (CD) and Earth Mover's Distance (EMD) are used to calculate the reconstruction loss. To modify the generated samples, interpolation and latent space arithmetic are used. While these techniques can be used to modify samples generated by all different latent representation models (AEs, GANs, etc.), they only allow control over the existence of an attribute and not the desired shape. Also, direct part modification is not possible since there is only a global latent code that controls the shape with an entangled representation.

For part editing and generation, the most popular approach is reconstructing or generating the parts separately by different networks and then assembling them to form the global shape by an additional composition network. In [4], a "Spatial Transformer Network" is used to combine the generated parts by applying affine transformations. CompoNet [17] uses a separate Encoder-Decoder model for each part. Encoders are used to get codes for each part and a composition network outputs transformation parameters per part. The generated parts are warped together using the transformation parameters. In [11], VAE-GANs (Variational Autoencoder Generative Adversarial Networks) are used to generate parts instead of naive AEs. VAE-GAN uses a Variational Autoencoder instead of a Generative network, so it is an Encoder-Decoder-Discriminator architecture. In [18], an inverse approach is adopted where a low-resolution global shape is generated first and then a part refiner module enhances the generated parts by refining and completing the missing regions. Most of these studies use voxels as input data because of the ease of data processing. Most part based studies assume that different parts have the same number of points.

StructureNet [12] is one of the pioneer studies for part editing and generation. It uses two encoders and two decoders, one to process geometry and one to process relations between parts with graph networks. While the results are very detailed, the model requires training with fine-grained and hierarchical part annotations, which is not always available. We designed our system to work with a simple labeling indicating to which part a point belongs to. Also we expect from our system to learn the relations between parts without specifically trained for it since it operates on latent space for semantic modifications.

The studies in the literature use multiple neural networks with different architectures to solve the problem of shape generation with respect to parts. The parts are generated independently and then they are processed by scaling, positioning and rotating to form a meaningful global shape. We aim to solve the problem with a single neural network that can handle part-aware global shape generation without any need for additional processing to form a meaningful global shape. The disentangled latent space allows exchanging and removal of existent parts or generation of new parts that fits the global model. Part generation is an intermediate step of the main process that results in global shape generation. The proposed method provides a holistic approach that generates the global shape with respect to part semantics instead of generating the parts separately. The proposed method can work on unannotated point clouds with the additional segmentation ability. The simplicity of the approach allows using a smaller model with fewer parameters than previous studies.

3 Proposed Method

The proposed method is an end-to-end system consisting of 3 modules: Feature extractor, Segmentation and Decoder which are explained in Sections 3.1, 3.2, and 3.3 respectively. A generative module can also be integrated to provide generative capabilities which is explained in Section 3.4.

3.1 Feature Extractor

The feature extractor is based on a modification of the standard PointNet architecture and introduces a part feature extraction step between the point feature extraction module and the global symmetric function

(Fig.1). The point feature extractor is a multi-layer perceptron (MLP) model that takes n points and outputs l features for each point. PointNet applies max-pooling on the first axis to get the global feature. Max-pooling is a symmetric function and it gives the same result for the same input in any order so it is invariant to permutations of the input set. In the proposed method, instead of directly applying a global max-pooling, max-pooling is applied on a part to get an individual part feature. After this step, max-pooling is applied again on these part features to obtain the global feature for the whole shape. The idea is based on a 2-stage max-pooling operation which can be defined as max of maxes similar to the "reduce max" operation in parallel programming. Directly applying max operation on a vector of numbers gives the same result as applying the operation in multiple iterations. In this context, the first max operation is used to get part features and the subsequent max is used to get the global feature. In this way, while obtaining the same global feature as the original network, a number of separate part features are also obtained. This operation is shown in Eq. 2 where h is approximated by MLP and symmetric function g is max-pooling.

$$\begin{aligned} f_{p=1,...,k}(\{x_1,...,x_n\}) \approx & g_{p=1,...,k}(h(x_1),...,h(x_n)) \\ f_s(\{x_1,...,x_n\}) = & \, g(f_{p=1},...,f_{p=k}) \\ f : 2^{\mathbb{R}^N} \rightarrow \mathbb{R}, h : \mathbb{R}^N \rightarrow & \mathbb{R}^L, g : \mathbb{R}^L \times ... \times \mathbb{R}^L \rightarrow \mathbb{R} \end{aligned} \quad (2)$$

Each point in the point cloud has three variables defining its spatial position in $\mathbb{R}^3$. The MLP component extracts l features from each point, resulting in a $n \times l$ point feature vector. Feature extractor is fed with the point cloud data and the part labels are extracted by the segmentation module. The part feature extractor applies max-pooling on each part separately ($g_{p=1,...,k}$), by taking the part labels into account and produces k separate part feature vectors ($f_{p=1,...,k}$), each having a size of l. Then a $k \times l$ vector is formed by concatenating these vectors together. The global feature extractor applies global max-pooling g to produce a global feature f_s of size l. By this way, k individual part features, in addition to a global feature, are obtained. The part features can be modified individually to change the part only or the global feature can be modified to change the global shape. This allows modification of specific parts, in addition to the modification of global shapes.

3.2 Segmentation

The part feature extractor needs part labels to generate part features. In part-segmented point cloud datasets, for a model with k parts (For example, a chair model has $k = 4$ semantic parts; seat, back, arm and leg), represented with n points, there are n labels, associating each point with a part label. While there are part labels in annotated datasets, such information is rarely available in real conditions. Segmentation module is employed to segment the unlabeled point clouds to get part labels. It uses point features generated by the point feature extractor to generate per-point part labels. Then these labels are fed to the part feature extractor. During the training, the segmentation module is trained together with the system using the ground truth part labels from the training data. During inference, the segmentation module generates the part labels, eliminating the need for ground-truth part labels and making the system an end-to-end solution for unannotated point clouds.

As an alternative to end-to-end training with the whole system, the module can be trained in isolation or can be trained using a pretrained point-wise feature extractor. All training options generate similar results withing a range of 2% with respect to segmentation performance. The point features can be concatenated with global features to improve the segmentation performance, allowing segmentation by considering local and global features together. This method decreases segmentation loss by around 50% over using the point features only. The global features are extracted by a max operation on point features.

The aim of the segmentation module is to predict part labels when they are not available. If the part labels are available, then this module can be omitted and these labels can directly be fed into the part feature extractor. This makes the reconstruction performance better as expected since the part labels are not predictions but ground truths. While this is a better option for reconstruction performance, it eliminates the ability of the system to work with unannotated raw point clouds.

3.3 Decoder

The aim of the decoder is to generate a $n \times 3$ point cloud from the global feature vector l. An MLP or a Deconvolutional model can be employed for this purpose. The decoder is trained with reconstruction loss to enforce reconstruction of a given sample with the minimum loss. Decoder learns to generate corresponding global shapes for given global feature vectors. Modified feature vectors are fed to the decoder to get the modified point cloud models. Segmentation module can be used for segmenting the generated samples if necessary.

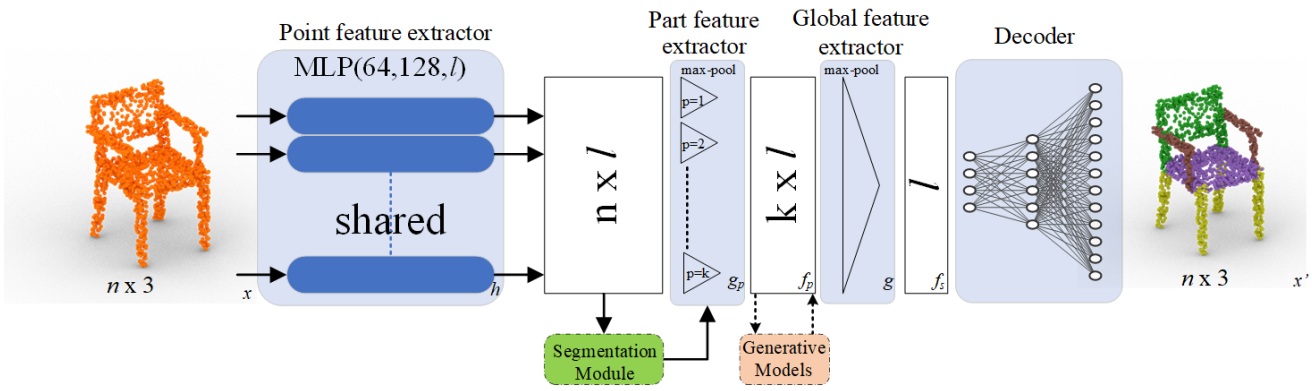

Fig. 1: The proposed architecture consists of a point-wise feature extractor, a part feature extractor, a global feature extractor and a decoder. The optional generative model allows generation of new parts and models. The optional segmentation module allows the system to work with unlabeled data.

3.4 Generative capabilities

The proposed method has an inherent capability to form new shapes by part feature exchange and by combining different part features. In addition, it allows integration of generative models to generate completely new parts and shapes. For this purpose, we created two variants using two different generative models: latent-space GAN (l-GAN) and VAE. l-GAN model and VAE sampling layers were integrated in between the part feature extractor and the global feature extractor to expand the system to have part generation ability -in addition to its ability to generate the global shape-.

Latent-space GAN (l-GAN) [1] works in latent space instead of the actual data space. A naive GAN is placed between the Encoder and Decoder that takes part features of the dataset as real input and aims to generate fake part features that result in realistic shapes when decoded. While GAN can generate novel and realistic samples, training may become unstable in the long run, resulting in mode collapse. Also GAN suffers from lack of diversity in generated samples. WGAN [2] proposes a better objective function using Wasserstein distance to address these problems:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[D(x)] - E_{z\sim p_z(z)}[D(G(z))] \quad (3)$$

It has also been implemented to work in the latent space (l-WGAN) to observe the differences. Gradient penalty has been applied and Discriminator has been trained more for more stable training [7].

While there are different AE implementations for point clouds based on PointNet, VAE based ones may fail because of the imbalance between regularization and reconstruction quality. Such models suffer from poor reconstruction/poor generation capabilities [1]. To overcome the imbalance problem, an additional coefficient β is used to weigh the regularization term. The objective function of VAE can be defined using a variational lower bound as:

$$\mathcal{L} = \mathbb{E}_{q_\phi(z|x)}[log p_\theta(x|z)] - \beta D_{KL}(q_\phi(z|x)||p(z)) \quad (4)$$

where q and p are data projection and generation modules with parameters ϕ and θ respectively and D_{KL} is Kullback–Leibler divergence [10].

4 Experimental Evaluation

Dataset: We used re-organized ShapeNetPart dataset [19], which is a subset of the highly popular ShapeNet 3D dataset. It contains part labels for more than 16000 models in 16 categories and the number of parts for each category varies from 2 to 6. Each point in the point cloud sample has a semantic part label. From these 16 categories, chair, table, plane, and car categories have been used for the study since they have the highest number of samples (3746, 5266, 2690 and 1824 samples, respectively). Each sample has a different number of points varying from 500 to 3000 points. For all the experiments, 2048 points per sample have been used, unless otherwise stated. To set all the samples the same size, random down-sampling or zero-padding have been applied. Parts can have any number of points for each model. Training, validation and test sets have been constructed with a 90%, 5%, 5% split. PyTorch has been used for implementation and PyTorch3D has been used for 3D operations [15]. The training took around an hour on an NVIDIA RTX2070 GPU for the base model. Code is publicly available at https://github.com/cihanongun/LPMNet

Distance metrics: Chamfer distance (CD) and Earth Mover's Distance (EMD) are the most commonly used metrics to measure the similarity of point clouds and compute the reconstruction error [5]. Both these metrics are permutation invariant and work on unordered sets. Chamfer Distance is a nearest neighbor distance metric for point sets. It is the squared distance of a point in the first set to the nearest neighbor point in the second set. Chamfer Distance between two point clouds S_1 and S_2 is defined as:

$$d_{CD}(S_1, S_2) = \sum_{p_1 \in S_1} \min_{p_2 \in S_2} \|p_1 - p_2\|_2^2 + \sum_{p_2 \in S_2} \min_{p_1 \in S_1} \|p_1 - p_2\|_2^2 \quad (5)$$

Earth Mover's Distance (EMD) [16] (a.k.a. Wasserstein Metric) is an algorithm to measure the effort to transport one set to another. EMD for two equal-sized point clouds S_1 and S_2 is defined as:

$$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \to S_2} \sum_{p \in S_1} \|p - \phi(p)\|_2 \quad (6)$$

where ϕ is a bijection. While in practice, the exact computation of EMD is prohibitively expensive, an approximate method with reported approximation error around 1% has been used [5].

The Base model: The AE architecture is inspired from [1]. The feature extractor is a PointNet model consisting of 3 1-D convolutional layers with kernel size 1 and feature size (64, 128, l). Each layer is followed by a ReLU activation function and a batch normalization layer. Input and feature transform subnetworks are omitted since the samples are already aligned. It has been observed that the original 5-layer architecture has no advantage over the proposed model even with more features. The segmentation module is a 4-layer MLP (64, 32, 16, k) with weight sharing and a softmax function at the end and it is trained with a classification loss. A 3-layer architecture gives similar performance with less overfitting but the performance drops with increasing feature size. Higher number of layers cause overfitting as the data is not complex and the proposed model is trained with single class. However, a more sophisticated architecture can be employed for more complex input data. The decoder generates the point cloud model with 3 fully connected layers (1024, 2048, $n \times 3$) and the first two layers are followed by a ReLU function. Fewer number of layers fail to generate high quality samples while models with higher number of layers tend to overfit to training data. A model with deconvolutional layers is also a viable option. A 5-layer (512, 256, 256, 128, 3) deconvolutional architecture has similar performance to the base model with less overfitting. However, deconvolutional model is sensitive to feature size and it fails when feature size is high (e.g. 1024). For the base model, the feature size l is 128 and number of points n is 2048. The system has been trained using Chamfer distance as reconstruction loss and cross-entropy loss as segmentation loss. Adam optimizer [9] has been used with a learning rate of 5×10^{-4} for 1000 epochs.

Experiment design: To evaluate the proposed method, we have conducted a number of experiments similar to those in the literature and introduced new ones. Unless otherwise stated, the base model has been used in all experiments. Evaluation of the reconstruction performance is provided in Section 4.1, followed by the evaluation of new model generation performance in Section 4.2. The proposed method has been tested with different input sizes to prove its robustness against low-resolution data and missing points and the results are provided in Section 4.3.

4.1 Evaluation of Reconstruction

We first evaluated the effect of different feature (bottleneck) sizes. Fig. 2 shows the reconstruction losses calculated using Chamfer and EMD for different feature sizes for the chair category. The proposed method and the baseline method [1] exhibit a similar trend that both suffer from higher reconstruction loss when the feature size is less than 128. In addition, to evaluate the effect of the part feature extractor on the reconstruction quality, the proposed part feature extractor has been integrated into the baseline method [1]. The results show no significant difference, supporting our claim that the global feature is not affected by the part feature extraction step. According to Fig. 2, a feature size of 128 provides a good balance to run the system with a smaller feature space without sacrificing reconstruction performance; so the feature size is set to 128 for all experiments.

The reconstruction results on the test set can be seen in Fig. 3. Visual results indicate good reconstruction performance with minor loss.

Part interpolation and part exchange experiments aim to validate that a regularized part feature space can extract the part features separately and parts can be exchanged between different generated shapes. Then, we show that different parts from different shapes can be used to compose new shapes.

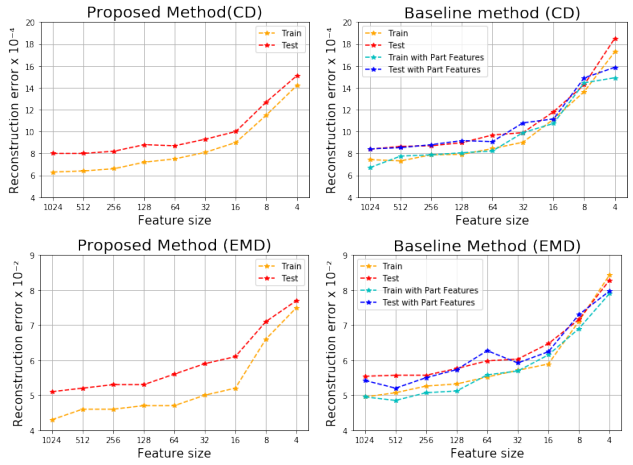

Fig. 2: The reconstruction losses for different feature sizes.

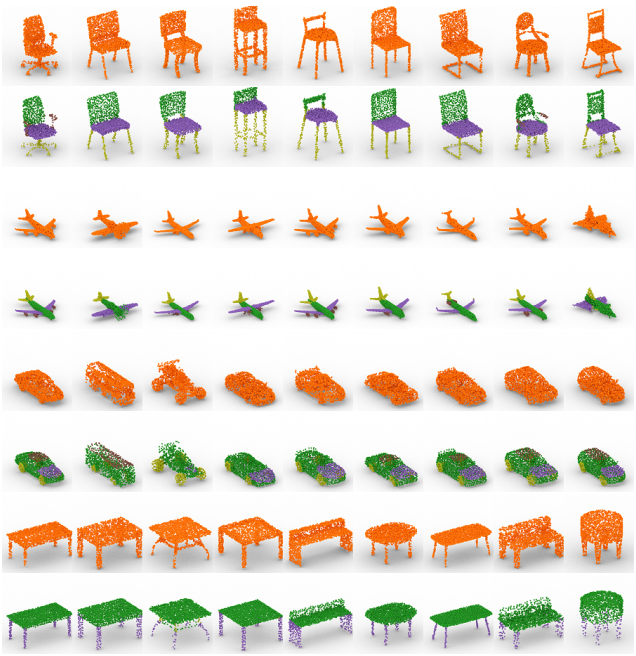

Fig. 3: The reconstruction results of the proposed model. For each object class, the first row shows the samples from the unlabeled test set and the second row shows the corresponding reconstructions.

Part interpolation and part exchange: By modifying the part feature, shape of a respective part could be changed in isolation, keeping the other parts the same. To prove this claim, we apply part interpolations for all parts separately and show the results in Fig. 4. Global feature interpolation results in a smooth interpolation between two different shapes reflecting a regular and continuous latent space. Part feature interpolation interpolates only a specific part and assembles the new part into the existing sample. Here it can be seen that it is not a naive part assembly transplanting a part into another shape. Latent space represents the semantic properties of a part so it generates a part that matches better to the new shape by preserving semantic properties. For example, using the leg part feature of a four-legged chair with an office chair having wheels generates the same office chair with four legs instead of wheels. However, the leg part will not be the same as the source chair since it would not be a good fit for the target office chair. The office chair is now generated with four legs which are in better harmony with the rest of the shape resulting in a more realistic looking chair. Results for other classes can be seen in Fig. 8.

Composition of separate parts: In the proposed architecture, the parts are expected to be independent of the global shape. To test the validity of the independence assumption of the parts, different part features from different models are merged to obtain a global feature. This global feature is then used to generate a global shape with these parts. Sample results can be seen in Fig. 5. A new shape is formed by the selected parts without any need for assembling the parts together with affine transformations. It has to be noted that the parts are not exactly the same as they are in source shapes. This is because the new shape is formed using the semantic properties of these parts. The experiments validate that new samples can be generated using different parts from different shapes.

4.2 Evaluation of New Model Generation

The method can be extended to have generative capabilities by integration of generative models. In this section, we evaluate the generation of new global shapes and parts by integrating two separate models: GAN and Variational Autoencoder (VAE).

Latent-space GAN based architecture [1] uses encoded data as its input and output. Generator is a 3-layer MLP (128, l, $k \times l$) for k parts and the Discriminator mirrors the Generator. Generator input is a 128-dimensional vector sampled from a Normal distribution. l-GAN has been trained using Adam optimizer with a first-moment value of 0.5 and learning rates of 5×10^{-4} and 1×10^{-4} for Generator and Discriminator respectively. GAN has been trained with the pretrained model to extract and decode features. WGAN follows the same architecture with a different objective function.

VAE based architecture follows the base model with an exception of the sampling layers, which are now fully connected layers to generate mean and sigma values.

Fig. 4: Part interpolation between 2 shapes. The first row is global shape interpolation between 2 shapes (most left and most right). Other rows are single part interpolations. Only corresponding part feature is interpolated while others are kept as same. This provides part modification and part exchange abilities between models.

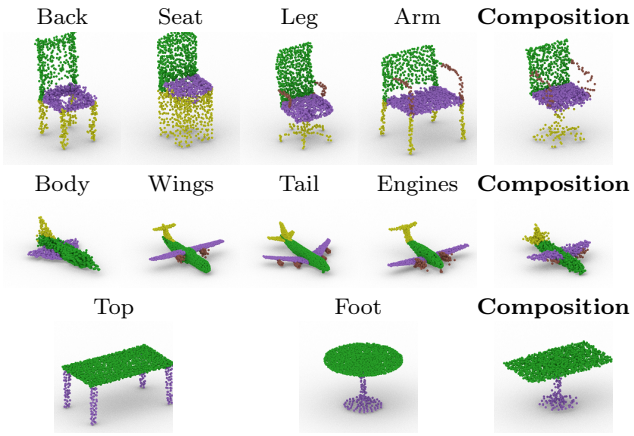

Fig. 5: Part features from different samples are combined together to form a new shape. Parts preserve semantic properties while fitting to the new shape.

Regularization term has been normalized with input dimension and β parameter has been set to 0.1 since it provides a good balance between reconstruction and generation quality. Reparametrization trick has been employed and the system has been trained using Adam optimizer [9] with a learning rate of 10^{-3} for 10000 epochs. For new data generation, latent codes have been sampled from a Normal distribution. Generated samples can be seen in Fig. 6 for chair class and Fig. 9 for plane, car and table classes.

For the evaluation of generative models, we have used the following metrics: Coverage (Cov), Minimum Matching Distance (MMD) and Jensen–Shannon Divergence (JSD) [1]. Cov measures the representation of a point cloud set S_2 in set S_1. It is the fraction of point clouds in one set that is matched to others by finding the nearest neighbor. MMD is the average of distances between the matched point clouds in different sets. JSD is the distance between 2 probability distributions, it is derived from Kullback–Leibler divergence [10]. In this scope, it is used as a measure of occupation of similar locations in 3D coordinate space between two point cloud sets. MMD and Cov have been calculated using both CD and EMD.

New samples are generated by five different approaches: (*i*) *part feature exchange*: randomly exchanging part features between different samples, (ii) *part feature composition*: composing new shapes by combining different

Fig. 6: Samples from generative models. VAE provides good reconstruction and generation capabilities. While standard GAN is able to generate good results, it suffers from lack of diversity. WGAN generates more diverse results.

part features from different random samples, (*iii*) *VAE*: new shapes are generated by sampling from a Normal distribution using VAE, (*iv*) *GAN*: GAN is used after training to randomly generate new shapes, (*v*) *WGAN*: WGAN is used instead of GAN for more diversity and more stable training. All models have been trained with CD and EMD. Chair category has been used to generate a sample set 3 times the size of the test set. Results can be seen in Table 1. As expected, the results are in favor of the models trained with the same distance metric as the evaluation method. Part exchange has the lowest distance score with a high coverage. This is expected since only a single part per sample is different from the reference test set. Also, high coverage supports the similarity between the test set and the part-exchange set. The random part composition approach exhibits good diversity and novelty comparable with the generative models. VAE implementation exhibites overfitting and collapses to a single mode when trained with EMD distance. WGAN achieves better diversity as expected with better coverage scores than GAN. The results are comparable with the best baseline [1] model which is a Gaussian Mixture Model on the latent space learned by an AE. The results show that different alternatives are successful at different aspects and they may serve different tasks better depending on the quality, diversity or complexity requirements of a particular task.

Table 1: Evaluation of generative models based on Minimum Matching Distance (MMD), Coverage (Cov), and Jensen-Shannon Divergence (JSD). Both CD($\times 10^{-4}$) and EMD($\times 10^{-2}$) is used for evaluation. AE-GMM is the best model in the baseline [1] study.

Model	MMD (CD)	MMD (EMD)	% Cov (CD)	% Cov (EMD)	JSD
Trained with CD					
Exchange	13.39	9.3	73.43	24.21	0.045
Comp.	15.79	9.68	69.53	19.53	0.050
VAE	14.01	10.38	67.18	21.09	0.077
l-GAN	16.84	9.61	49.21	20.31	0.054
l-WGAN	16.11	9.49	67.18	28.12	0.058
Trained with EMD					
Exchange	15.66	6.38	66.40	78.12	0.004
Comp.	19.94	7.12	55.46	64.06	0.008
VAE	31.61	9.94	12.5	6.25	0.154
l-GAN	18.93	6.63	48.43	54.68	0.012
l-WGAN	20.22	7.02	59.37	67.96	0.012
AE-GMM[1]	20.00	6.50	68.9	67.4	0.020

4.3 Robustness Against Different Input Sizes

The same shape can be defined by using different number of points. So, the method is expected to have the ability to process different input point cloud sizes (resolutions) and give similar outputs. In this section, we evaluate the performance of the proposed method against different input sizes and compare the critical points extracted from different input sizes.

To define a global feature, a feature extractor first detects the critical points, which are the most important points in a point cloud sample. The critical point set is the minimum number of points defining the shape. For example, the corner points are the critical points that define a triangle. The feature set defines the semantics of the shape irrespective of the resolution, so a higher resolution sample also results in the same global feature set (i.e., the corners of a triangle).

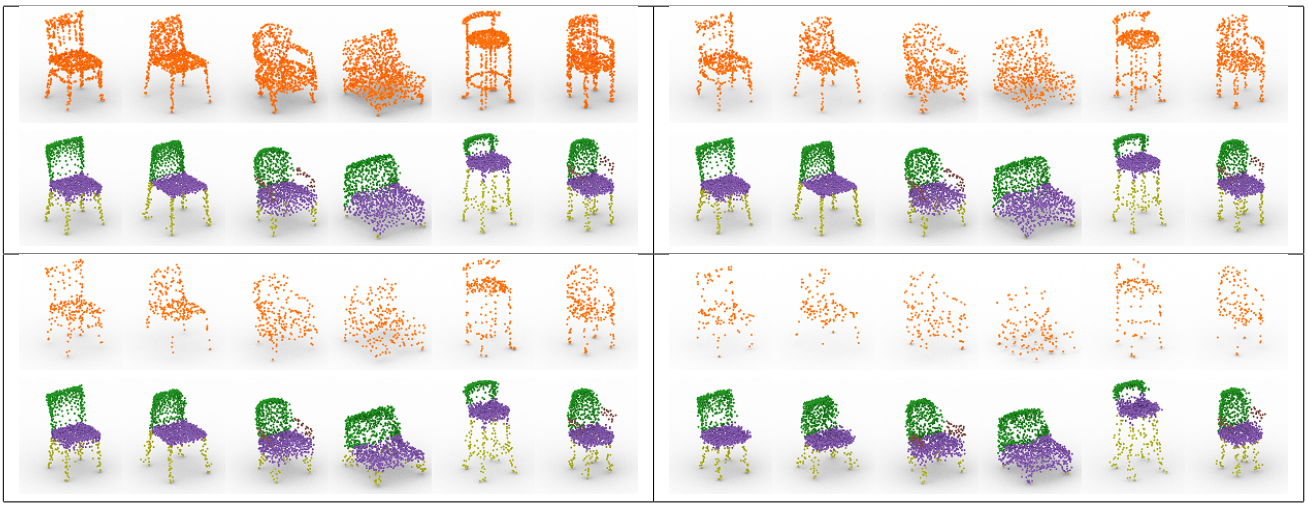

Fig. 7: Reconstruction results from 1024 (top-left), 512 (top-right), 256 (bottom-left) and 128 (bottom-right) points to 2048 points.

The proposed method is expected to extract the same feature set for a shape defined with different number of points. These features can then be decoded to reconstruct the shape at any size. To test this, the original input has been randomly downsampled to 1024, 512, 256 and 128 points from 2048 points. Then these samples have been zero-padded to obtain 2048 points and the zero-padded points have been labeled as part 0. Then, these samples have been fed into the pretrained network to reconstruct the shape. Since the network ignores part 0 for feature extraction, it extracts the same features for all input dimensions. The results in Fig. 7 shows that the system can handle different input dimensions by giving the same features for the same shapes. The results are not affected by the lack of zero-padded samples during training. Also, this approach can serve as an upsampling network without training from scratch. It has to be noted that a lower number of input points result in poorer reconstructions since some critical points vanish due to random downsampling. Removing batch normalization layers improves robustness with more independent point features.

5 Conclusions

In this paper, a generic part-aware architecture allowing exchanging of parts between different models and generating new point cloud models and parts has been proposed. The proposed system is based on a single network and does not need separate networks for each part or an additional network to assemble them to form a new shape. The system has been proven to work with different object categories having different numbers of parts and varying sizes. Also with the integrated segmentation module, the system provides an end-to-end solution for unlabeled data. It has been shown that GANs and VAEs can be integrated into the proposed method to generate new parts and models.

In the proposed method, while a part feature represents the corresponding part in a global shape, the decoder takes a global feature as input and outputs a global shape. While the method cannot reconstruct the parts separately, this is not considered to be a significant limitation as the ultimate aim in most applications is to form a global shape. To reconstruct the parts separately, the method must be trained with parts separately from scratch. Then, the global shape can be constructed from the parts by a composition model similar to those in the literature. Part modification and generation are complementary operations to get the global shapes.

In some cases, reconstruction of uncommon samples (e.g., asymmetrical samples, samples with incorrect labels) may fail, especially if they are only encountered in the test set. These samples are considered to be outliers by the network and they have limited effect in the learning and hence they are not represented effectively by the network. Processing outliers is a common and challenging problem for neural networks based systems.

Acknowledgements This work has been supported by Middle East Technical University Scientific Research Projects Coordination Unit under grant number GAP-704-2020-10071.

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: International Conference on Learning Representations (ICLR) (2018)
2. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 214–223 (2017)
3. Charles, R.Q., Su, H., Kaichun, M., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 77–85 (2017). DOI 10.1109/CVPR.2017.16
4. Dubrovina, A., Xia, F., Achlioptas, P., Shalah, M., Groscot, R., Guibas, L.J.: Composite shape modeling via latent space factorization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 8140–8149 (2019)
5. Fan, H., Su, H., Guibas, L.: A point set generation network for 3d object reconstruction from a single image. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2463–2471 (2017). DOI 10.1109/CVPR.2017.264
6. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems (NIPS), pp. 2672–2680 (2014)
7. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: Advances in neural information processing systems, pp. 5767–5777 (2017)
8. Kingma, D., Welling, M.: Auto-encoding variational bayes. In: International Conference on Learning Representations (ICLR) (2014)
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015)
10. Kullback, S., Leibler, R.A.: On information and sufficiency. Ann. Math. Statist. **22**(1), 79–86 (1951). DOI 10.1214/aoms/1177729694
11. Li, J., Niu, C., Xu, K.: Learning part generation and assembly for structure-aware shape synthesis. arXiv preprint arXiv:1906.06693 (2019)
12. Mo, K., Guerrero, P., Yi, L., Su, H., Wonka, P., Mitra, N., Guibas, L.J.: Structurenet: Hierarchical graph networks for 3d shape generation. ACM Trans. Graph. **38**, 242:1–242:19 (2019)
13. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 165–174 (2019)
14. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Neural Information Processing Systems (2017)
15. Ravi, N., Reizenstein, J., Novotny, D., Gordon, T., Lo, W.Y., Johnson, J., Gkioxari, G.: Accelerating 3d deep learning with pytorch3d. arXiv:2007.08501 (2020)
16. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. International Journal of Computer Vision **40**(2), 99–121 (2000)
17. Schor, N., Katzir, O., Zhang, H., Cohen-Or, D.: Componet: Learning to generate the unseen by part synthesis and composition. In: The IEEE International Conference on Computer Vision (ICCV) (2019)
18. Wang, H., Schor, N., Hu, R., Huang, H., Cohen-Or, D., Huang, H.: Global-to-local generative model for 3d shapes. ACM Transactions on Graphics (Proc. SIGGRAPH ASIA) **37**(6), 214:1—214:10 (2018)
19. Yi, L., Kim, V.G., Ceylan, D., Shen, I.C., Yan, M., Su, H., Lu, C., Huang, Q., Sheffer, A., Guibas, L.: A scalable active framework for region annotation in 3d shape collections. SIGGRAPH Asia (2016)

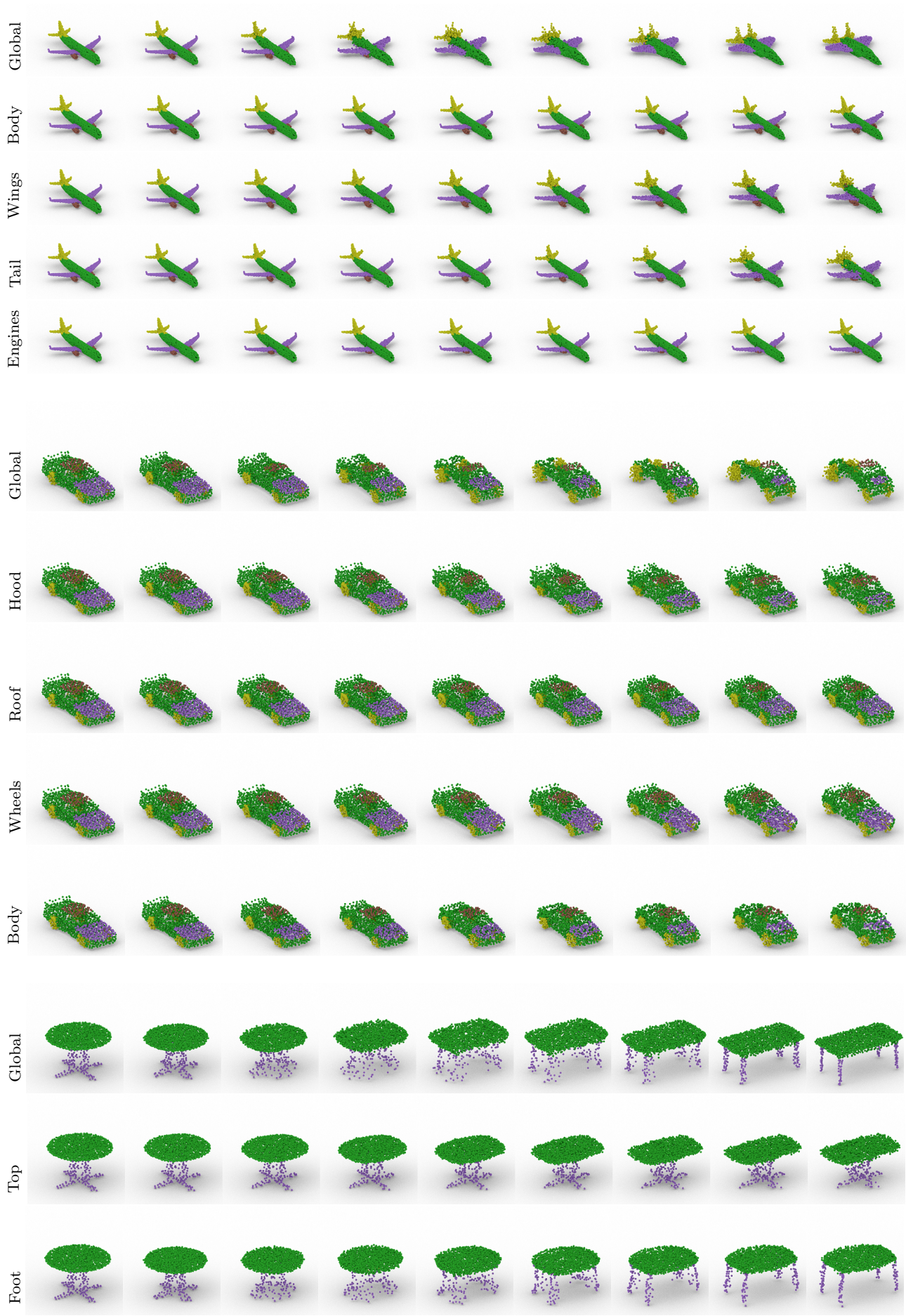

Fig. 8: Part interpolation results for plane, car and table classes.

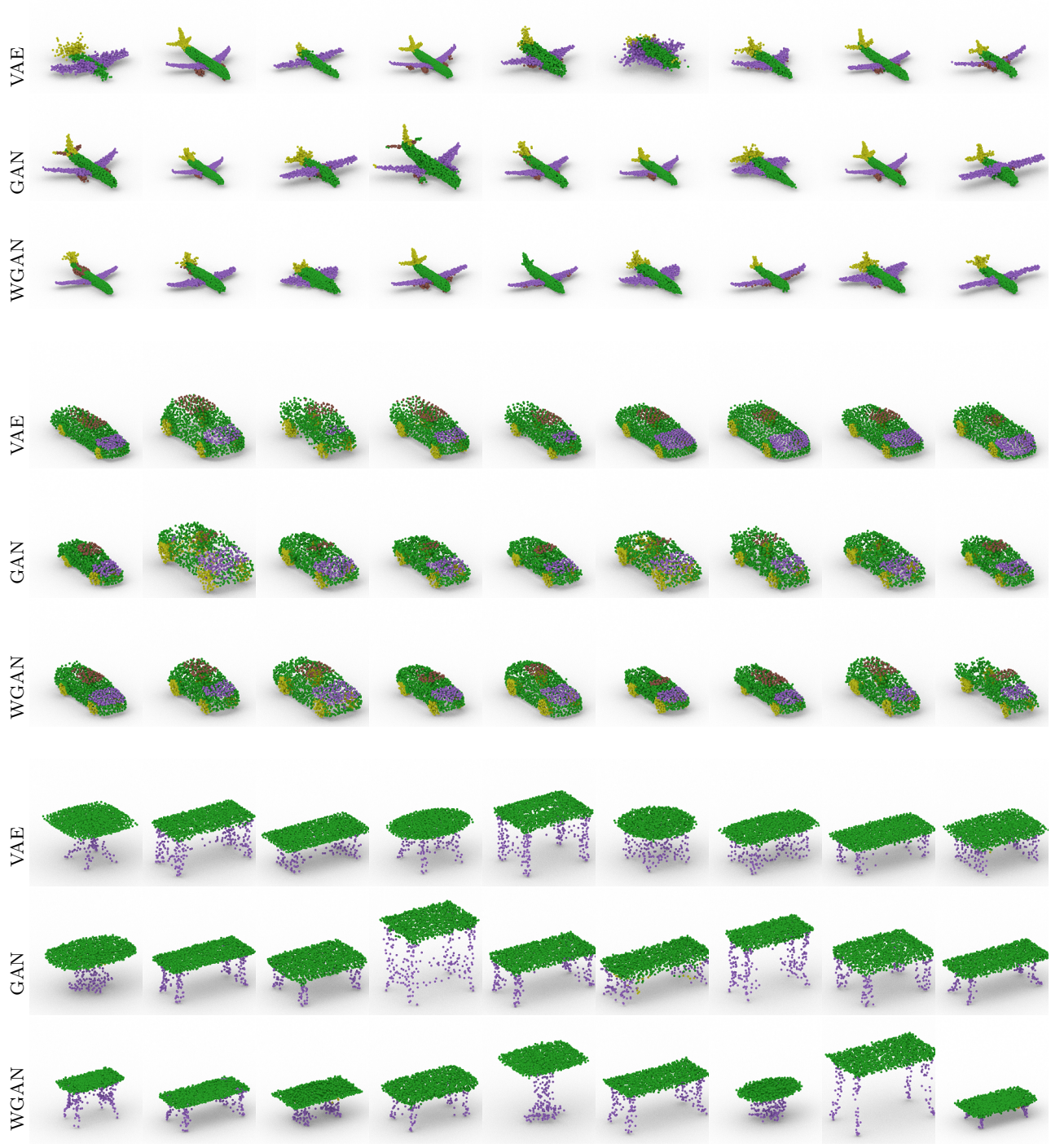

Fig. 9: Samples from generative models for plane, car and table classes.