Deformed Implicit Field: Modeling 3D Shapes with Learned Dense Correspondence

Yu Deng*[1,2] Jiaolong Yang[2] Xin Tong[2]
[1]Tsinghua University [2]Microsoft Research Asia
{t-yudeng,jiaoyan,xtong}@microsoft.com

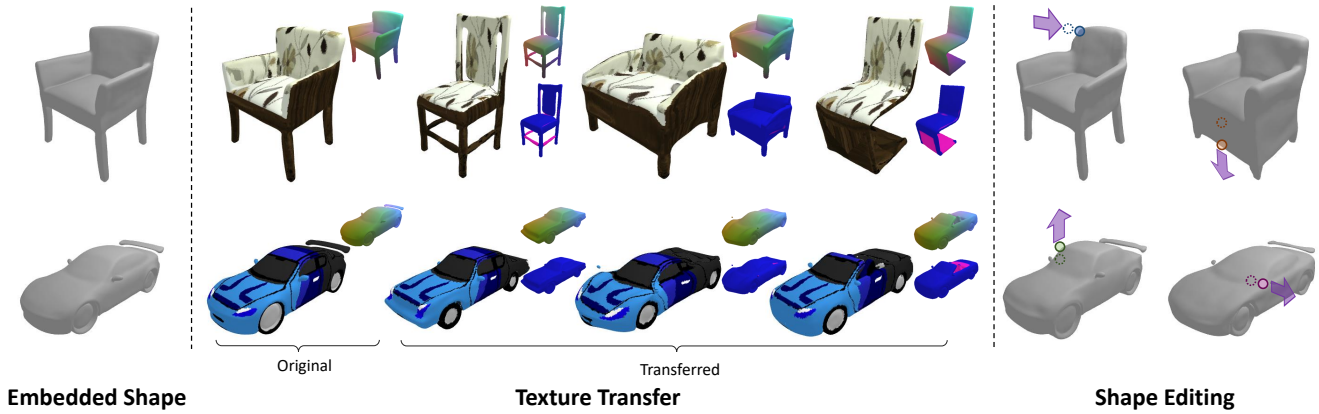

Figure 1. Our DIF-Net can produce 3D shapes with dense correspondences for object categories containing complex geometry variation and structure differences. It enables high-quality texture transfer shown in the middle four columns, where the two smaller figures after each transfer result show the color-coded correspondences (top) and their uncertainty (bottom; blue and red indicates low and high uncertainty respectively). With our learned shape space and correspondence, shapes can be freely edited by simply moving one or a sparse set of points, as shown in the last two columns.

Abstract

We propose a novel Deformed Implicit Field (DIF) representation for modeling 3D shapes of a category and generating dense correspondences among shapes. With DIF, a 3D shape is represented by a template implicit field shared across the category, together with a 3D deformation field and a correction field dedicated for each shape instance. Shape correspondences can be easily established using their deformation fields. Our neural network, dubbed DIF-Net, jointly learns a shape latent space and these fields for 3D objects belonging to a category without using any correspondence or part label. The learned DIF-Net can also provides reliable correspondence uncertainty measurement reflecting shape structure discrepancy. Experiments show that DIF-Net not only produces high-fidelity 3D shapes but also builds high-quality dense correspondences across different shapes. We also demonstrate several applications such as texture transfer and shape editing, where our method achieves compelling results that cannot be achieved by previous methods. [1]

1. Introduction

3D objects in a same class share some common shape features and semantic correspondences, which can be used to construct a deformable shape model beneficial for a diverse array of downstream tasks in 3D and 2D domains such as shape understanding [31, 2], reconstruction [49, 10, 60], manipulation [8, 26], and image synthesis [50, 54, 48].

Learning a 3D shape model with dense correspondences is a longstanding task in computer vision and graphics. However, existing works mostly focus on object classes with consistent geometric topologies such as human face and body [8, 53, 32, 35, 61]. Shapes in these object categories can be pre-aligned for 3D model construction. Recent deep learning based approaches directly learn a latent space of 3D objects [55, 46, 1, 39]. Although these methods can model complex objects, they do not deal with dense correspondences between 3D shapes.

In this paper, we investigate learning model of 3D shapes and their dense correspondences for more generic objects such as cars and chairs. Compared to human face and body, these object classes exhibit much larger shape variations and structure changes, rendering correspondence con-

*This work was done when Yu Deng was an intern at MSRA.
[1]Code URL: https://github.com/microsoft/DIF-Net.

struction extremely challenging. For these object categories, even human cannot reliably label the dense correspondences between two arbitrary shapes.

To achieve this goal, we act on recent advances in deep implicit fields, which have shown extraordinary power of representing complicated 3D geometry [37, 39, 14, 42], and propose a novel Deformed Implicit Field (DIF) representation for joint shape latent space and dense correspondence learning. With DIF, a 3D shape is represented by a template implicit field, shared across the category, together with a 3D deformation field and a scalar correction field, dedicated for each shape instance. The output implicit field of a shape can be constructed by deforming the template implicit field and applying correction. The deformation field serves as a shape alignment function, with which dense correspondence between two shapes can be established by deforming their surfaces to the aligned 3D space. The correction field is introduced to enhance shape representation capability.

We apply a neural network called DIF-Net to learn these fields together given a collection of shapes. To achieve unsupervised correspondence learning without any label, our key observation is that the normal direction of a shape point is highly correlated to its semantic information and very useful for correspondence reasoning. In light of this, we simply enforce the normals of two corresponding surface points connected by deformation to be close. In addition, we impose a spatial smoothness constrain on the deformation fields and enforce the correction fields to be minimal. Thorough qualitative and quantitative evaluations show that our DIF-Net trained in this way can produce high quality correspondences. Moreover, correspondence uncertainty reflecting structure discrepancy between two shapes can be reliably measured by our method.

The contribution of this paper can be summarized as follows: **1**) we propose DIF, a novel implicit field based 3D shape representation modeling shape deformations across an object category; **2**) we propose DIF-Net, the first method devoted to 3D shape modeling with dense correspondences learned in an unsupervised fashion for objects with structure variation; **3**) we show that our method can achieve high-quality dense correspondences and compelling texture transfer and shape editing results that cannot be achieved by previous methods. We believe our method can be applied in a wide range of 3D shape analysis and manipulation tasks.

2. Related Work

3D Shape Models with Correspondence. Building 3D shape model for a class of shapes has been actively studied in the past. Perhaps the most famous 3D shape model is the 3D morphable model (3DMM) introduced by Blanz *et al*. [8] for human faces. To build a 3DMM model, face scans are aligned by shape registration methods to derive correspondences, based on which shape deformation bases can be obtained via PCA. The 3DMM model has brought a profound impact to human face related research [41, 59, 51, 28, 33, 49, 21, 20, 18, 50, 48, 16]. Apart from face, Loper *et al*. [35] build a 3D morphable model for skinned human body which can control body shapes and poses. This model has been applied in various tasks such as pose estimation [9, 3] and image manipulation [54]. Similarly, Zuffi *et al*. [61] propose a morphable model for animals. The object categories handled by these methods typically have consistent topologies where shapes can be aligned to build correspondences. They do not address more complex object classes containing structure variations.

Learning Shape Latent Space. A large volume of methods [55, 46, 1, 7, 6, 25, 58, 45, 19, 39] have been proposed in recent years to model 3D shapes and learn a latent shape space using deep neural networks, especially generative adversarial networks (GANs) [24] and variational autoencoders [30, 45]. However these methods do not explicitly model the dense correspondence among different shapes. Our method not only learns a shape latent space but also generates dense shape correspondence.

Implicit Shape Representation. Recent studies show that learning implicit functions for 3D shapes excels at representing complicated geometry [39, 37, 14, 23, 22, 4, 5, 42, 17]. For example, Park *et al*. [39] use a neural network to approximate the signed distance field (SDF) of 3D shapes and show superior results compare to voxel and mesh based representations [46, 25]. Sitzmann *et al*. [42] show that surfaces of complex scene can be represented by a simple 5-layer MLP with periodic functions as activation. However, these methods mainly target at high-fidelity surface reconstruction and cannot reveal shape correspondence. Genova *et al*. [23] introduce an implicit template constructed with multiple RBF kernels. They can obtain a coarse dense correspondence between shapes by deforming and relocating RBF kernels to fit different shapes. However, their learning process is designed only for shape reconstruction thus the obtained correspondences are not reliable. Our representation in this paper is also based on implicit fields, but enables correspondence reasoning. Our new loss functions leads to high-quality correspondences learned without any label.

Structured Shape Representation. Structured representations are also widely used to model complex shapes with varying structures [52, 15, 44, 12, 38, 19, 40, 13]. By decomposing 3D shapes into small parts, a complicated shape can be represented by primitive elements such as cuboids [52, 44], superquadrics [40], convexes [15, 12], and RBF kernels [23]. Many of these methods can provide part-level correspondences among shapes, but do not model dense correspondences.

3. Approach

3.1. Overview

Given a collection of 3D objects $\{\mathcal{O}_i\}$ from one category, our goal is to learn a latent shape space $\mathcal{L}$ as well as a neural shape model f that can generate these objects and provide dense shape correspondence. Each shape can be represented by a latent code $\alpha \in \mathbb{R}^k$ in $\mathcal{L}$, and the shape model f maps the latent code to corresponding 3D shape,

$$f : \alpha \in \mathbb{R}^k \rightarrow \mathcal{O} \tag{1}$$

with a neural network. We adopt the auto-decoder framework presented in [39] to jointly learn the shape codes $\{\alpha_j\}$ for the given objects and the weights of model f. This auto-decoder framework can give rise to a decent latent space as shown in [39]. We can also apply proper latent space regularization as in VAE training [30] to learn a desired latent space, which will be discussed later.

After training, new shapes can be generated by latent space sampling, and a shape can be embedded into the latent shape via inverse optimization.

Implicit field. To generate high-fidelity shapes, we use signed distance fields (SDF) which can faithfully represent surface geometry details using a neural network as the field function [39]. SDF is a continuous representation which assigns any point $p \in \mathbb{R}^3$ a scalar value $s \in \mathbb{R}$:

$$SDF(p) = s, \tag{2}$$

where the magnitude of s represents the distance from p to its closest shape surface and the sign indicates whether p is inside (negative) the shape or outside (positive). With an SDF, shape surface can be implicitly represented by the iso-surface of $SDF(\cdot) = 0$. A 3D mesh can be extracted from this implicit surface using off-the-shelf algorithms such as Marching Cubes [36]. Using SDF to represent shapes, our neural shape model can be rewritten as

$$f : (\alpha, p) \in \mathbb{R}^{k+3} \rightarrow s \in \mathbb{R}. \tag{3}$$

Network components. The task in Eq. (3) intertwines shape information decoding from latent codes and SDF prediction for 3D points. Inspired by [43], we employ two networks to decompose this task: a *DIF-Net* for SDF prediction and a *Hyper-Net* for shape information decoding, as shown in Fig. 2. Hyper-Net Ψ predicts the weights ω for the neurons in DIF-Net Φ, and the two networks can be written as

$$\Psi : \alpha \in \mathbb{R}^k \rightarrow \omega \in \mathbb{R}^m, \tag{4}$$

$$\Phi_\omega : p \in \mathbb{R}^3 \rightarrow s \in \mathbb{R}. \tag{5}$$

Our DIF-Net Φ consists of two sub-networks: a template SDF generation network T and a Deform-Net D. We will introduce these two sub-networks and our deformed implicit field representation in the next section.

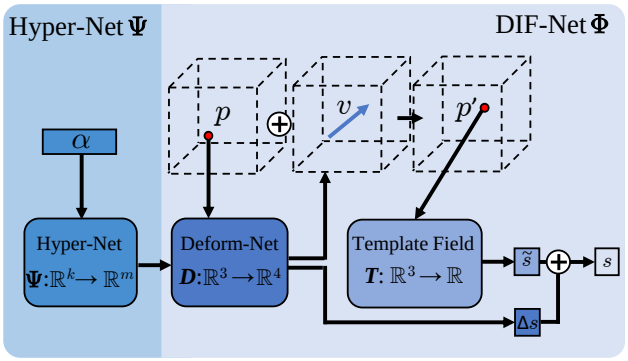

Figure 2. Overview of our proposed method. For a shape code α, Hyper-Net Ψ predicts (a part of) the weights of DIF-Net Φ, which further predicts the SDF for the shape. DIF-Net Φ consists of Deform-Net D which predicts a 3D deformation field and a correction field for the shape, and network T for generating a template implicit field shared across all shapes.

3.2. Deformed Implicit Field Representation

For a given object class, we assume that the object instances are mostly composed by a few common patterns or semantic structures. This is a mild assumption valid for many real-world object classes. For example, all cars consist of bodies and tires, and most chairs have back, seat, and legs. We seek to find a template implicit field which depicts common structures of the class and can derive the implicit fields for different object instances through 3D deformation.

Template implicit field and deformation field. To obtain an optimal template implicit field, we jointly learn it with deformations during training. Our template SDF generation network T and Deform-Net D can be written as

$$T : p \in \mathbb{R}^3 \rightarrow s^t \in \mathbb{R}, \tag{6}$$

$$D_\omega : p \in \mathbb{R}^3 \rightarrow v \in \mathbb{R}^3. \tag{7}$$

The network weights of T is shared across the whole class, whereas weights of D are instance-specific and derived from the Hyper-Net, as shown in Fig. 2 and indicated by subscript ω in Eq. (7). With T and D_ω, the SDF value of point p can be obtained via

$$s = T(p + v) = T(p + D_\omega(p)). \tag{8}$$

With the predicted deformation fields $D_\omega(p)$, dense correspondences between two shapes can be established by deforming their surface to the aligned template space. In practice, one can densely sample surface points on two shapes, deforming them to the aligned 3D space, and find closest neighbors between two point sets.

Correction field. We also learn an SDF correction field to enhance our DIF representation. Specifically, we use Deform-Net D to predict not only a deformation vector but also a scalar correction term Δs,

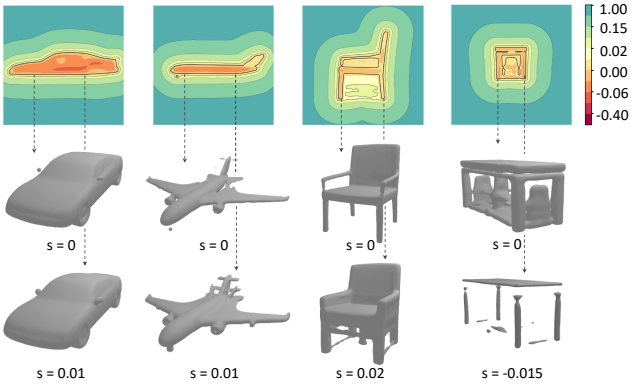

Figure 3. The learned template implicit fields (*Y-Z* slice) for four object categories and different iso-surfaces extracted from them. Note that the template implicit fields are not valid shape SDFs and they characterize different shape structures within a category.

$$D_\omega : p \in \mathbb{R}^3 \rightarrow (v, \Delta s) \in \mathbb{R}^4, \tag{9}$$

and we change the SDF value output in Eq. (8) to

$$s = T(p + v) + \Delta s = T(p + D^v_\omega(p)) + D^{\Delta s}_\omega(p). \tag{10}$$

As illustrated in Fig. 4, the correction term is helpful to generate desired shapes. We empirically found that this correction term is also important to learn high-quality correspondence, as we will show in the experiments.

In summary, our neural shape model can be written as

$$f(\alpha, p) = \Phi_{\Psi(\alpha)}(p) = T(p + D^v_{\Psi(\alpha)}(p)) + D^{\Delta s}_{\Psi(\alpha)}(p), \tag{11}$$

which is parameterized by the weights of network Ψ and T.

Discussions. It is crucial to note that *our template implicit field is fundamentally different from a template 3D shape*. In fact, Figure 3 shows that the learned template implicit fields are not valid shape SDFs (*e.g.*, compare the lower part of chair's template field slice and the extracted iso-surface at $s = 0$). Instead, they characterize different shape structures within a category. In the experiments, we find that if under some circumstances (*e.g.*, training with improper losses) they degenerate to a valid shape SDF representing a certain shape, both shape reconstruction accuracy and correspondence quality will significantly drop. This shows the advantage of our representation over previous works using template meshes or grids [25, 56].

3.3. Learning Shape and Correspondence with DIF

We use the auto-decoder framework presented in [39] to jointly train weights of networks Ψ and T and learn latent codes $\{\alpha_j\}$. We design new loss functions for DIF to learn desirable dense correspondences.

Given a collection of shapes, we first apply an SDF regression loss similar to [42] to learn the SDFs of these shapes. Let $\Phi_i(p)$ be the short-hand notation for $\Phi_{\Psi(\alpha_i)}(p)$

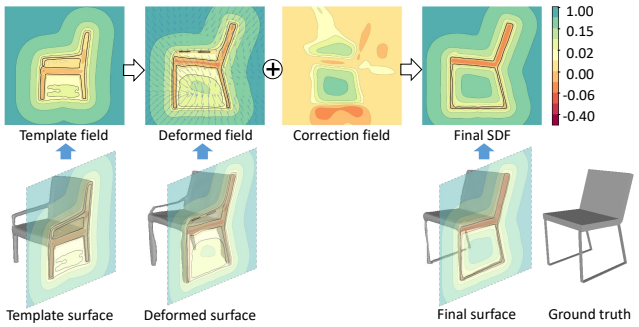

Figure 4. 2D and 3D visualization of our SDF prediction process.

which is the predicted SDF value, we have

$$\begin{aligned} L_{sdf} = \sum_i \Big(\sum_{p \in \Omega} |\Phi_i(p) - \bar{s}| + \sum_{p \in \mathcal{S}_i} (1 - \langle \nabla \Phi_i(p), \bar{n} \rangle) \\ + \sum_{p \in \Omega} |\|\nabla \Phi_i(p)\|_2 - 1| + \sum_{p \in \Omega \setminus \mathcal{S}_i} \rho(\Phi_i(p)) \Big), \end{aligned} \tag{12}$$

where $\bar{s}$ and $\bar{n}$ denote the ground-truth SDF value and surface normal respectively, ∇ denotes the spacial gradient of a 3D field, Ω is the 3D space and $\mathcal{S}_i$ denotes shape surfaces. In practice, points will be sampled in the free space and on shape surface to calculate the loss. The second term in Eq. (12) is used to learn correct normals on shape surfaces – the gradient function of an SDF equals the surface normal given surface points as input and can be easily computed using network backpropagration. The third term is derived from the Eikonal equation which enforces the norm of spatial gradients $\nabla \Phi_i$ to be 1. The last term penalizes SDF values close to 0 for non-surface points through $\rho(s) = \exp(-\delta \cdot |s|), \delta \gg 1$. We refer the readers to [42] for mode details about this loss. As in [39], we also apply a regularization loss to constrain the learned latent codes:

$$L_{reg} = \sum_i \|\alpha_i\|_2^2. \tag{13}$$

Alternatively, we can apply stronger regularization on the latent space, such as minimizing the Kullback–Leibler divergence between the latent code posterior distribution and Gaussian distribution as in VAE training [30]. More details and results can be found in the *suppl. material*.

Normal consistency prior. To learn desired correspondences, our key observation is that the normal of a surface point is highly correlated with its semantic information. For example, normals on car hoods always point to the sky, and normals on the left doors always point to the left. In light of this, we encourage the normal directions of points in the template space to be consistent with their correspondences on all given shape instances:

$$L_{normal} = \sum_i \sum_{p \in \mathcal{S}_i} \Big(1 - \langle \nabla T\big(p + D^v_{\omega_i}(p)\big), \bar{n} \rangle \Big), \tag{14}$$

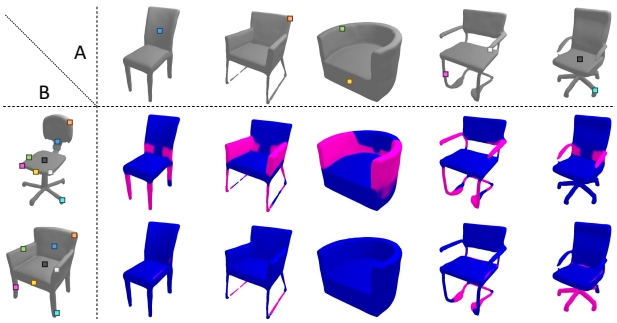

Figure 5. Correspondence uncertainty visualization. Each figure in the bottom right sector shows the uncertainty of shape A (top row)'s correspondence found on B (left column). Red and blue color represents high and low uncertainty, respectively. We also draw some colored points on A and their correspondences with the same color on B.

where ∇T is the spatial gradient of template field T, and $\bar{n}$ denotes the ground-truth normal of point p on object surface $\mathcal{S}_i$. Since the template field is used to derive all final shapes through deformation, this loss essentially enforces *the normal consistency for correspondences across all the shapes* generated by our network. Note that this loss is different with the normal term in Eq. (12): generating correct normals for each shape, as enforced by the latter, does not necessitate consistent correspondence normals between shapes, which is yet imposed by the former.

Deformation smoothness prior. To encourage smooth deformation and avoid large shape distortion, we add a simple smoothness loss on the deformation field:

$$L_{smooth} = \sum_i \sum_{p\in\Omega} \sum_{d\in\{X,Y,Z\}} \|\nabla D^v_{\omega_i}|_d(p)\|_2, \quad (15)$$

which penalizes the spatial gradient of the deformation field along X, Y and Z directions.

Minimal correction prior To encourage shape representation through implicit field deformation rather than correction, we minimize the correction field via

$$L_c = \sum_i \sum_{p\in\Omega} |D^{\Delta s}_{\omega_i}(p)|. \quad (16)$$

In summary, the whole training process can be formulated as the following optimization problem:

$$\arg\min_{\{\alpha_j\},\Psi,T} L_{sdf} + w_1 L_{normal} + w_2 L_{smooth} + w_3 L_c + w_4 L_{reg} \quad (17)$$

where w's are the balancing weights for different loss terms.

3.4. Correspondence Uncertainty Measurement

In practice, it is desirable to have a quality or uncertainty metric for the obtained shape correspondences, which can be used for structure difference analysis, bad correspondence removal *etc*. As mentioned previously, the correspondence between two objects $\mathcal{O}_i$ and $\mathcal{O}_j$ can be built by nearest neighbor search in the template space. Let p_i be a point on $\mathcal{O}_i$ and p_j its corresponding points found on $\mathcal{O}_j$, we propose a simple yet surprisingly-effective uncertainty metric based on their distance in the template space:

$$u(p_i, p_j) = 1 - \exp(-\gamma \|(p_i + v_i) - (p_j + v_j)\|_2^2) \quad (18)$$

where $v_i = D^v_{\omega_i}(p_i)$ is the deformation vector (similarly for v_j) and γ is a scaling factor. The examples in Fig. 5 show that the regions with high uncertainty computed by Eq. (18) conform well to structure discrepancy between shapes.

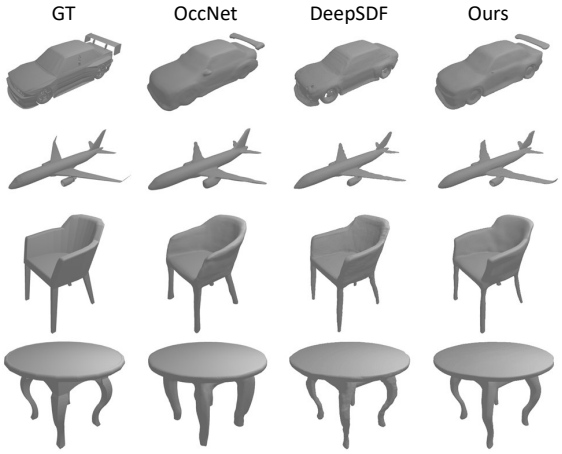

Figure 6. Shape reconstruction results for unseen shapes by OccNet [37], DeepSDF [39] and our DIF-Net.

4. Experiments

Implementation Details We implement Hyper-Net Ψ as ten MLPs to predict the weights of Deform-Net layers. Each MLP has a single hidden layer of dimension 256 and ReLU activations. The Deform-Net D and the template field network T are both MLPs with three hidden layers of dimension 128, equipped with sine activations advocated by [42]. We train our model on four categories in ShapeNet-V2 [11], including *car*, *airplane*, *chair*, and *table*. We use the first 3K, 3.5K, 4K, and 4K instances in the dataset as training set for the above categories, respectively. All parameters are trained end-to-end using the Adam [29] optimizer with learning rate $1e{-}4$ for 60 epochs. Training takes about 4 hours on 8 NVIDIA V100 GPUs with batchsize 256 for one category. Due to space limitation, more details are presented in the *suppl. material*.

4.1. Ability of Shape Representation

To evaluate the representation power of DIF and the latent shape space learned by DIF-Net, we embed new shapes *unseen* in the training stage and measure the reconstruction accuracy. With our trained networks Ψ and T, we embed a test shape to the latent space by solving the following simplified optimization problem of Eq. (17):

$$\arg\min_{\alpha} L_{sdf} + w_4 L_{reg}. \quad (19)$$

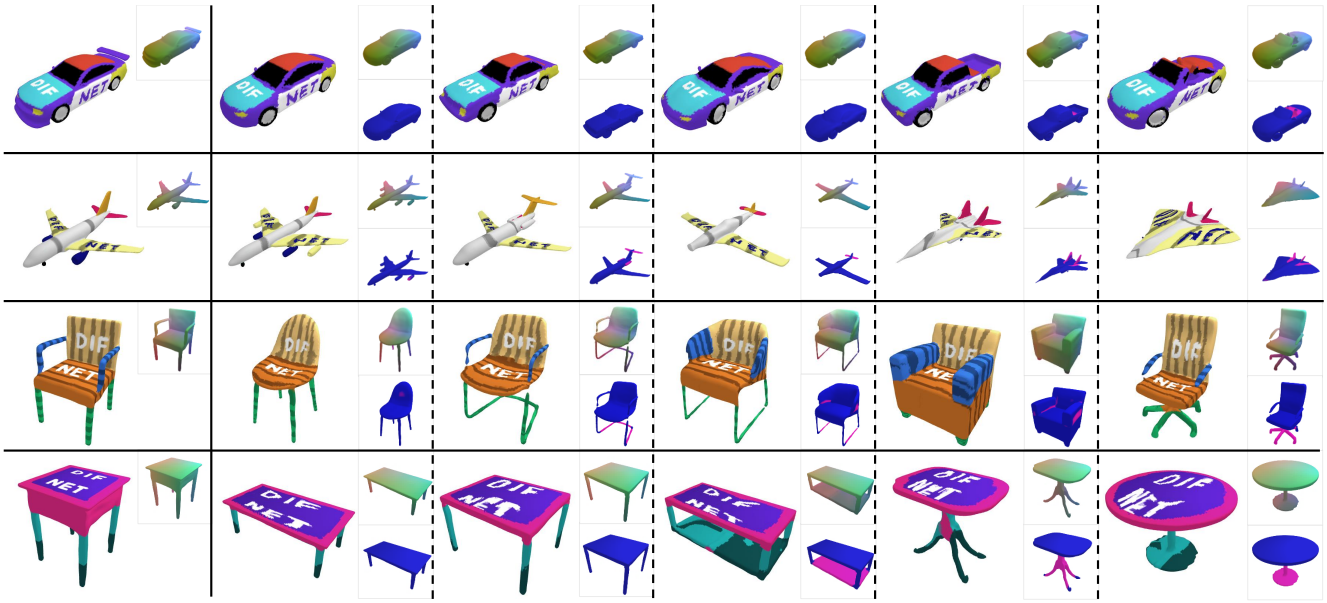

Figure 7. Qualitative evaluation of our learned dense correspondence for each category. For better visualization, we manually paint a generated shape (first column) with different colors on different semantic components. We also draw some strip patterns and texts on the shapes. Then we transfer these colors to other shapes generated by our method (last five columns) according to their correspondences. We also visualize the correspondences color-coded by spatial coordinates as well as the correspondence uncertainty for each shape.

CD ($\times$1000)	car	plane	chair	table
OccNet* [37]	0.582	0.288	0.995	1.326
DeepSDF† [39]	0.767	0.298	0.785	1.422
Ours	**0.404**	**0.249**	**0.661**	**1.036**
Ours w/o deform.	0.353	0.255	0.529	0.772
EMD	car	plane	chair	table
OccNet* [37]	0.037	0.025	0.045	0.047
DeepSDF† [39]	0.041	0.029	0.038	0.046
Ours	**0.036**	**0.024**	**0.038**	**0.040**
Ours w/o deform.	0.034	0.025	0.036	0.037

Table 1. Reconstruction accuracy for unseen shapes. We use the first 100 shapes in the intersection of the test set splits from DeepSDF [39] and ours. OccNet [37] is trained for each category using our training data, and a per-category DeepSDF model trained by [34] is evaluated here. Reconstructed meshes are extracted at a resolution of 256^3 for all methods. CD and EMD are evaluated using 10K and 8K sampled points respectively.

We compare with two state-of-the-art shape modeling methods based on deep implicit filed: OccNet [37] and DeepSDF [39]. For OccNet, we train an individual model for each category using our training data for a fair comparison. For DeepSDF, we use a per-category model trained by [34] to evaluate its performance.

Table 1 shows the shape reconstruction accuracy on 100 test shapes for each category, measured with chamfer distance (CD) and earth mover distance (EMD), and Fig 6 visually compares some results. It can be seen that all three methods perform well in representing unseen shapes, and our method is slightly better in terms of numerical error.

We also compare with a variant of our method which does not model dense correspondence. Specifically, we replace DIF-Net Φ with a MLP of three hidden layers that directly predicts the SDF of a shape. The numerical results are presented in Table 1, which are slightly better than our DIF-Net. It indicates that the deformation-based implicit design only leads to moderate decease of representation capability. However, high-quality dense correspondences can be achieved with this design, as we will show in the following sections.

Due to space limitation, more evaluations of our trained DIF-Net including **latent space interpolation**, **sampling** and **retrieval** are deferred to the *suppl. materials*.

4.2. Learned Dense Correspondence

Qualitative evaluation. Figure 7 visualizes the correspondences generated by our method, where we manually paint salient color patterns on the shapes to better check the correspondence quality. Visually inspected, our method produces convincing correspondences across various shapes despite their structure differences. It not only correctly matches shared semantic components between two shapes but also preserves the original color patterns. Moreover, the uncertain regions revealed by our method well reflect structure differences between two shapes.

Quantitative evaluation via Label Transfer. To our knowledge, there is no dataset offering ground truth dense correspondence for objects with structure variation. There-

Figure 8. Labeled points on the original shapes are deformed into the template 3D space where the semantic parts are well aligned

IoU	car	plane	chair	table	average
Closest Point	62.7 | 63.5	60.5 | 62.5	65.9 | 69.6	68.5 | 73.4	64.4 | 67.2
Atlas-sph. [25]	62.6 | 64.0	51.1 | 50.9	56.7 | 57.9	64.0 | 67.0	58.6 | 59.9
Atlas-25 [25]	59.3 | 60.3	54.2 | 52.2	62.1 | 64.9	66.2 | 69.1	60.5 | 61.6
SIF [23]	62.6 | 63.9	52.3 | 52.3	57.6 | 57.0	65.7 | 68.7	59.6 | 60.5
Ours	**72.7** | **74.1**	**71.7** | **78.4**	**75.3** | **79.7**	**81.1** | **87.9**	**75.2** | **80.0**

Table 2. Label IoU (mean\|median) on label transfer task. For each category, we use 5 labeled source shapes and test the segmentation accuracy on our whole training set containing 3K-4K shapes. Note that AtlasNet [25] and SIF [23] are trained for each category individually for a fair comparison.

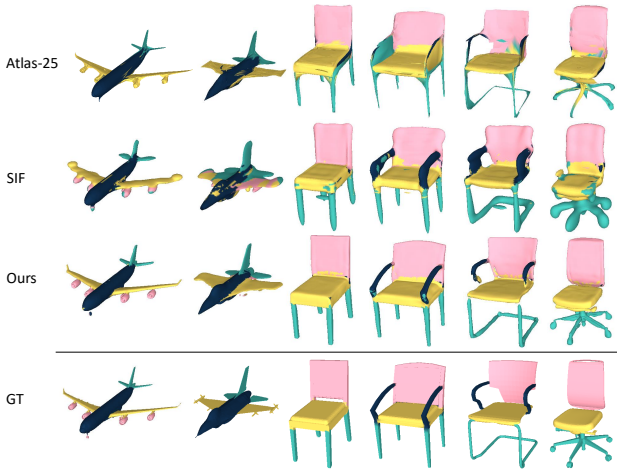

Figure 9. Quatitative comparison on label transfer task.

fore, we resort to a semantic label transfer experiment for quantitative evaluation. We use the ShapeNet-Part dataset [57] which contains part labels for ShapeNet objects. For each of the four object categories, we selected 5 labeled shapes as source shapes, and transfer their labels to other shapes leveraging dense correspondences. This task can be viewed as few-shot 3D shape segmentation learning using 5 samples as training data.

For this task, we first deform all labeled points on the 5 source shapes into our template 3D space, as shown in Fig. 8. For an unlabeled shape, we deform its surface points into the template 3D space, find 10 nearest labeled points for each, and then conduct simple label voting.

We compare our method with AtlasNet [25] and SIF [23], which can be used to build correspondences, as well as a naive closest point based method. For fair comparison, we train AtlasNet and SIF for each category individually with our training data. For label transfer with AtlasNet, we first label its template grid vertices which are

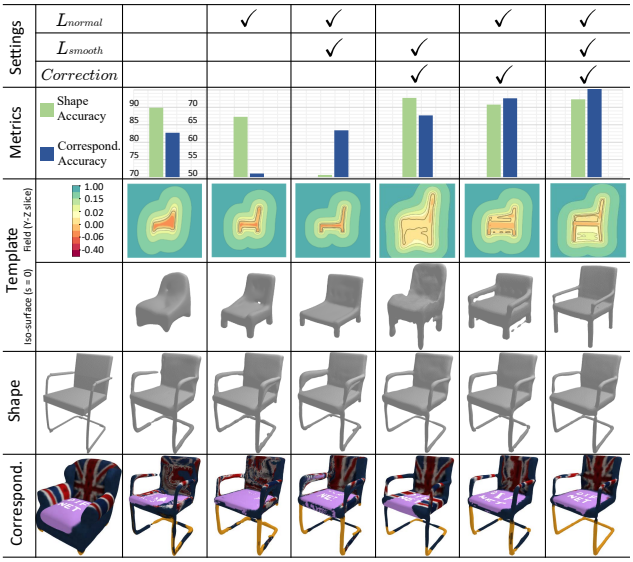

Figure 10. Influence of different training losses and the correction field in our model. We use the chair category for evaluation, and measure shape reconstruction accuracy via F-score [47] at $\tau = 0.001$ and correspondence accuracy via label IoU in the part label transfer task of Section 4.2. We also present visual results to further illustrate the effect of different components.

further used to label new points. For SIF, we follow a similar step to [23] to obtain template coordinates for all points to transfer labels. In the naive method, we label each point by direct label voting using its nearest labeled points in the original shape space. We use 10 nearest neighbors for all methods. More details can be found in the *supp. material.*

Table 2 compares the accuracy measured by IoU between ground-truth and transferred labels, and Fig. 9 shows some visual results. Our method outperforms all others by a wide margin. AtlasNet and SIF may generate inconsistent correspondences (*e.g.*, a point labeled as chair "arm" corresponds to "arm" regions for some shapes but to "seat" for others). Their results are worse than the naive closest point method for categories with large shape variation.

4.3. Ablation Study

In this section, we conduct ablation study to validate the efficacy of our training loss terms and the correction field. The main results of different settings are shown in Fig. 10. As can be seen, without the normal loss L_{normal}, the learned correspondences are inferior as indicated by the significant IoU drop in label transfer. Without the deformation smoothness loss L_{smooth}, the learned correspondences are highly distorted, as shown in the transferred textures. Without the correction field, shape representation ability of the model decreases significantly. Interestingly, Fig. 10 shows that the correction field is also crucial to obtain high quality correspondences. We hypothesis that with-

Figure 11. Texture transfer result on ShapeNet objects using correspondences generated by our DIF-Net. (**Best viewed with zoom**)

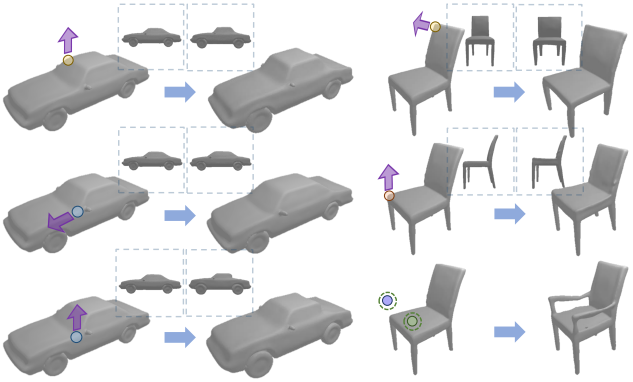

Figure 12. Shape editing result. DIF-Net can deform shapes and add new structures using only sparse points as guidance.

out the correction field, more complex deformation fields are needed to represent the final shape with various structures, which significantly increases the learning difficulty. Under this situation, learning may get stuck into local minima, leading to an inferior template field lacking rich structural information (as shown by the iso-surfaces in Fig. 10), which further results in correspondence accuracy drop. The correction field helps to learn a better template field, hence improving the correspondence quality.

5. Applications

5.1. Texture Transfer

Using dense correspondence generated by DIF-Net, we are able to transfer textures from one object to another. Successfully transferring rich textures among various shapes necessitates high quality dense correspondence. Figure 11 shows texture transfer results between ground truth shapes in ShapeNet. Visually inspected, the rich texture patterns are well preserved and transferred to correct semantic areas in new shapes. Figure 1 contains two more texture transfer results of our method for embedded 3D shapes.

5.2. Shape Editing

With the learned latent shape and dense correspondence, our method can be used to manipulate 3D shapes by moving one or a sparse set of points. Specifically, give a shape with embedded latent code α, we can freely select one 3D point p_1 on the shape and specify its desired new position p_2. Let $p_1' = p_1 + D^v_{\Psi(\alpha)}(p_1)$ be the deformed point of p_1 in the template 3D space, we achieve shape editing via solving for a new shape code $\hat{\alpha}$ minimizing the following equation:

$$\arg\min_{\hat{\alpha}} \|(p_2+v) - p_1'\|_2^2 + |\Phi_{\Psi(\hat{\alpha})}(p_2)| + \|\hat{\alpha}-\alpha\|_1 \quad (20)$$

where $v = D^v_{\Psi(\hat{\alpha})}(p_2)$ is the deformation vector for p_2 with the new shape code $\hat{\alpha}$. In this equation, the first term enforces the original and new points on the shapes before and after editing to be a correspondence pair thus having same semantic meaning. The second term ensures the new point lies on the new shape surface. The third term requires the code change to be small. Figure 12 shows the editing results for two shapes and another two examples can be found in Fig. 1. We can even add new structures to a given shape via Eq. (20), where in this case we directly select p_1' in the template space and a free point p_2 in the shape space. An example is shown in Fig. 12. More details and results regarding shape editing can be found in the *suppl. material*.

6. Conclusion

We have presented Deformed Implicit Field, a novel implicit-based representation modeling a class of 3D shapes and providing dense correspondences. We also presented DIF-Net, a neural shape model that learns high-quality dense correspondences in an unsupervised manner through our proposed loss functions. Various experiments and applications collectively demonstrated the high quality shapes and correspondences generated by our method. In future, we plan to extend the DIF representation to handling more generic 3D objects and scenes.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *International Conference on Machine Learning*, pages 40–49, 2018. 1, 2

[2] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. DensePose: Dense human pose estimation in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7297–7306, 2018. 1

[3] Anurag Arnab, Carl Doersch, and Andrew Zisserman. Exploiting temporal context for 3D human pose estimation in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3395–3404, 2019. 2

[4] Matan Atzmon and Yaron Lipman. SAL: Sign agnostic learning of shapes from raw data. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2565–2574, 2020. 2

[5] Matan Atzmon and Yaron Lipman. SAL++: Sign agnostic learning with derivatives. *arXiv preprint arXiv:2006.05400*, 2020. 2

[6] Timur Bagautdinov, Chenglei Wu, Jason Saragih, Pascal Fua, and Yaser Sheikh. Modeling facial geometry using compositional vaes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3877–3886, 2018. 2

[7] Heli Ben-Hamu, Haggai Maron, Itay Kezurer, Gal Avineri, and Yaron Lipman. Multi-chart generative surface modeling. *ACM Transactions on Graphics*, 37(6):1–15, 2018. 2

[8] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *26th Annual Conference on Computer Graphics and Interactive Techniques*, pages 187–194, 1999. 1, 2

[9] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J Black. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In *European Conference on Computer Vision*, pages 561–578. Springer, 2016. 2

[10] Chen Cao, Derek Bradley, Kun Zhou, and Thabo Beeler. Real-time high-fidelity facial performance capture. *ACM Transactions on Graphics*, 34(4):1–9, 2015. 1

[11] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. ShapeNet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015. 5

[12] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. BSP-Net: Generating compact meshes via binary space partitioning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 45–54, 2020. 2

[13] Zhiqin Chen, Kangxue Yin, Matthew Fisher, Siddhartha Chaudhuri, and Hao Zhang. BAE-Net: Branched autoencoder for shape co-segmentation. In *IEEE International Conference on Computer Vision*, pages 8490–8499, 2019. 2

[14] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 2

[15] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. CvxNet: Learnable convex decomposition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 31–44, 2020. 2

[16] Yu Deng, Jiaolong Yang, Dong Chen, Fang Wen, and Xin Tong. Disentangled and controllable face image generation via 3D imitative-contrastive learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5154–5163, 2020. 2

[17] Yueqi Duan, Haidong Zhu, He Wang, Li Yi, Ram Nevatia, and Leonidas J Guibas. Curriculum DeepSDF. In *European Conference on Computer Vision*, 2020. 2

[18] Yao Feng, Fan Wu, Xiaohu Shao, Yanfeng Wang, and Xi Zhou. Joint 3D face reconstruction and dense alignment with position map regression network. In *European Conference on Computer Vision*, pages 534–551, 2018. 2

[19] Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao Zhang. SDM-NET: Deep generative network for structured deformable mesh. *ACM Transactions on Graphics*, 38(6):1–15, 2019. 2

[20] Pablo Garrido, Michael Zollhöfer, Dan Casas, Levi Valgaerts, Kiran Varanasi, Patrick Pérez, and Christian Theobalt. Reconstruction of personalized 3D face rigs from monocular video. *ACM Transactions on Graphics*, 35(3):1–15, 2016. 2

[21] Kyle Genova, Forrester Cole, Aaron Maschinot, Aaron Sarna, Daniel Vlasic, and William T Freeman. Unsupervised training for 3D morphable model regression. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8377–8386, 2018. 2

[22] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3D shape. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4857–4866, 2020. 2

[23] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *IEEE International Conference on Computer Vision*, pages 7154–7164, 2019. 2, 7, 13

[24] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 2

[25] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3D surface generation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 216–224, 2018. 2, 4, 7

[26] Xiaoguang Han, Chang Gao, and Yizhou Yu. DeepSketch2Face: a deep learning based sketching system for 3D face and caricature modeling. *ACM Transactions on Graphics*, 36(4):1–12, 2017. 1

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision*, pages 1026–1034, 2015. 12

[28] Guosheng Hu, Fei Yan, Chi-Ho Chan, Weihong Deng, William Christmas, Josef Kittler, and Neil M Robertson. Face recognition using a unified 3D morphable model. In *European Conference on Computer Vision*, pages 73–89, 2016. 2

[29] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 5, 12

[30] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014. 2, 3, 4

[31] Christoph Lassner, Javier Romero, Martin Kiefel, Federica Bogo, Michael J Black, and Peter V Gehler. Unite the people: Closing the loop between 3D and 2D human representations. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6050–6059, 2017. 1

[32] Tianye Li, Timo Bolkart, Michael J Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4d scans. *ACM Transactions on Graphics*, 36(6):194–1, 2017. 1

[33] Feng Liu, Ronghang Zhu, Dan Zeng, Qijun Zhao, and Xiaoming Liu. Disentangling features in 3D face shapes for joint face reconstruction and recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5216–5225, 2018. 2

[34] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. DIST: Rendering deep implicit signed distance function with differentiable sphere tracing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2019–2028, 2020. 6

[35] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. SMPL: A skinned multi-person linear model. *ACM Transactions on Graphics*, 34(6):1–16, 2015. 1, 2

[36] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Computer Graphics*, 21(4):163–169, 1987. 3

[37] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 2, 5, 6

[38] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 909–918, 2019. 2

[39] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 1, 2, 3, 4, 5, 6, 12

[40] Despoina Paschalidou, Ali Osman Ulusoy, and Andreas Geiger. Superquadrics revisited: Learning 3D shape parsing beyond cuboids. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 10344–10353, 2019. 2

[41] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3D face model for pose and illumination invariant face recognition. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 296–301, 2009. 2

[42] Vincent Sitzmann, Julien NP Martel, Alexander W Bergman, David B Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems*, 2020. 2, 4, 5, 12

[43] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3D-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*, pages 1121–1132, 2019. 3

[44] Dmitriy Smirnov, Matthew Fisher, Vladimir G Kim, Richard Zhang, and Justin Solomon. Deep parametric shape predictions using distance fields. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 561–570, 2020. 2

[45] Qingyang Tan, Lin Gao, Yu-Kun Lai, and Shihong Xia. Variational autoencoders for deforming 3D mesh models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5841–5850, 2018. 2

[46] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs. In *IEEE International Conference on Computer Vision*, pages 2088–2096, 2017. 1, 2

[47] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3D reconstruction networks learn? In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3405–3414, 2019. 7

[48] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhofer, and Christian Theobalt. StyleRig: Rigging stylegan for 3D control over portrait images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6142–6151, 2020. 1, 2

[49] Ayush Tewari, Michael Zollhofer, Hyeongwoo Kim, Pablo Garrido, Florian Bernard, Patrick Perez, and Christian Theobalt. MoFA: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In *IEEE International Conference on Computer Vision*, pages 1274–1283, 2017. 1, 2

[50] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2387–2395, 2016. 1, 2

[51] Luan Tran and Xiaoming Liu. Nonlinear 3D face morphable model. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7346–7355, 2018. 2

[52] Shubham Tulsiani, Hao Su, Leonidas J Guibas, Alexei A Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2635–2643, 2017. 2

[53] Daniel Vlasic, Matthew Brand, Hanspeter Pfister, and Jovan Popovic. Face transfer with multilinear models. In *ACM SIGGRAPH 2006 Courses*, pages 24–es. 2006. 1

[54] Chung-Yi Weng, Brian Curless, and Ira Kemelmacher-Shlizerman. Photo wake-up: 3D character animation from a single photo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5908–5917, 2019. 1, 2

[55] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016. 1, 2

[56] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. FoldingNet: Point cloud auto-encoder via deep grid deformation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215, 2018. 4

[57] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3D shape collections. *ACM Transactions on Graphics*, 35(6):1–12, 2016. 7

[58] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Josh Tenenbaum, and Bill Freeman. Visual object networks: Image generation with disentangled 3D representations. In *Advances in Neural Information Processing Systems*, pages 118–129, 2018. 2

[59] Xiangyu Zhu, Zhen Lei, Junjie Yan, Dong Yi, and Stan Z Li. High-fidelity pose and expression normalization for face recognition in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 787–796, 2015. 2

[60] Silvia Zuffi, Angjoo Kanazawa, and Michael J Black. Lions and tigers and bears: Capturing non-rigid, 3D, articulated shape from images. In *IEEE conference on Computer Vision and Pattern Recognition*, pages 3955–3963, 2018. 1

[61] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6365–6373, 2017. 1, 2

A. More Implementation Details

Data Preparation. To train DIF-Net via the SDF regression loss L_{sdf} defined in Eq. (12) in the main paper, we randomly sample points on shape surface and in the free space. Specifically, we follow a similar step as in [39] to normalize each ground truth mesh into a sphere with radius of 1/1.03. Then, for surface points, we render 100 virtual images for each normalized mesh using 100 virtual cameras regularly sampled on the unit sphere. Surface points are obtained via back-projecting the depth pixels from these virtual images. Normals of these surface points are obtained in a similar way from virtual normal images. This sampling strategy helps us get rid of inner structures of each mesh that are invisible from outside (e.g car seats).

For free-space points, we uniformly sample them within a cube of $[-1, 1]^3$, and calculate their distance to the nearest surface points sampled using the above strategy. To decide the sign of distance for a free-space point, we render it with the same virtual cameras used to obtain surface points, and check if its depth is smaller than the depth of a surface point falling into the same pixel. As long as it has a smaller depth value in any virtual image, it will be classified as an external point and get a positive sign. Otherwise, it will be classified as an internal point and get a negative sign.

In the end, we randomly sample 500K surface points along with normals and 500K free space points with their SDF values for each mesh.

Network Architecture. Figure I shows the structures of Hyper-Net Ψ and DIF-Net Φ used in our method. All networks are MLPs. The Hyper-Net consists of multiple MLPs each responsible for the weights ω_i of a single fully-connected layer i in the Deform-Net D_ω. The weights of Template Field T are shared across the class.

Training Details. We jointly learn all latent codes $\{\alpha_i\}$, the Hyper-Net Ψ, and the Template Field T using the losses in Eq. (17) in the main paper. Similar to [39], we initialize all latent codes using $\mathcal{N}(0, 0.01^2)$. The weights of Hyper-Net Ψ are initialized using [27]. The weights of Template Field T are initialized as in [42].

We train an individual model on each category for 60 epochs. At each iteration, we randomly select 4K surface points and 4K free-space points for each shape in a batch to calculate the losses in Eq. (17) and update learnable parameters. During each epoch, there are in total 200K surface points and 200K free-space points used for each shape.

We set the balancing weights for four terms in the SDF regression loss L_{sdf} in Eq. (12) to $3e3$, $1e2$, $5e1$, and $5e2$ respectively following [42]. w_1 and w_4 are set to $1e2$ and $1e6$ respectively for all categories. For w_2, we set it to 5, 2, 5, and 1 for *car*, *airplane*, *chair*, and *table* respectively. For w_3, we set it to $1e2$, $1e2$, $5e1$, and $1e2$ for each of the above categories, respectively. Models for all categories are trained using an Adam optimizer [29] with a learning rate of $1e-4$ and a batchsize of 256.

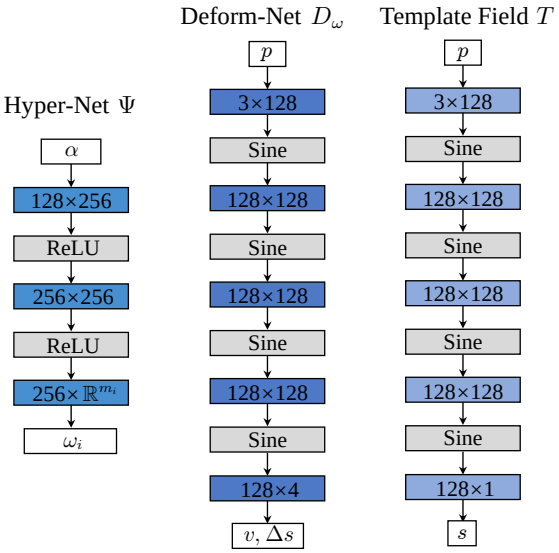

Figure I. Structures of different networks used in our method.

Inference Details. At inference time, given a test shape, we obtain its latent code via optimizing Eq. (19). Weights for different loss terms and point sampling strategy are identical to the training phase. We use Adam optimizer with a learning rate of $1e-4$, and update the latent code for 30 epochs in total.

B. Different Latent Code Regularization

In the main paper, we constrain the norm of learned latent code α to be small using L_{reg} defined in Eq. (13). Alternatively, we can also constrain the posterior distribution of latent code to be close to a Gaussian distribution to learn a latent space for better sampling. Specifically, we can replace the L_{reg} in Eq. (13) with the following loss:

$$L_{reg'} = KL(q(\alpha_i|\mathcal{O}_i)\|\mathcal{N}(\mu, \Sigma^\intercal\Sigma)) \tag{I}$$

where KL denotes the Kullback–Leibler divergence and $q(\alpha_i|\mathcal{O}_i)$ is the posterior distribution of α_i represented by a Gaussian distribution with mean equals to α_i. To calculate $L_{reg'}$ in Eq. (I), we have to obtain the standard deviation for $q(\alpha_i|\mathcal{O}_i)$ as well. Therefore, we introduce an extra learnable latent code σ with the same dimension of α to represent its standard deviation.

Specifically, at training stage, the input latent code α to the Hyper-Net Ψ is replaced by a random variable $\widetilde{\alpha}$ sampled from $\mathcal{N}(\alpha, \sigma^2 I)$. Then, we train all learnable parameters using the following losses:

$$\underset{\{\alpha_j\},\{\sigma_j\},\Psi,T}{\arg\min} \; L_{sdf} + w_1 L_{normal} + w_2 L_{smooth} + w_3 L_c + w_5 L_{reg'}, \tag{II}$$

Figure II. Source shapes of each category used for label transfer evaluation.

where $L_{reg'}$ is defined in Eq. (I) and all other losses are the same as in Eq. (17). In practice, we set the Gaussian distribution in Eq. (I) to $\mathcal{N}(0, 0.01^2)$, and set the weight w_5 to $1e2$. Other balancing weights are the same as described in the previous section.

C. Learned Shape Latent Space

Latent Space Interpolation. In this part, we show latent space interpolation results using the model trained with Eq. (17) in Fig. IV. Shapes in even positions are interpolated from their two neighbors. We also show the color-coded correspondence of them. As depicted, interpolated shapes are reasonable and correspondence between different shapes are consistent.

Latent Space Retrieval. Given a source shape, we can search for its nearest neighbors using the Euclidean distance between shape latent codes as metrics. Retrieval results are shown in Fig. V. Shapes on the right-hand side are nearest neighbors of the left-most shapes, and are sorted in an ascending distance order. Our learned latent space can capture the shape similarities in the original shape space—similar shapes are embeded close to each other.

Latent Space Sampling. We show shape sampling results using the model trained with Eq. (II) in Fig. VI. Our model can well capture the distribution of 3D shapes and generate new shapes.

D. More Details of Label Transfer

Source Shapes. Figure II shows the source shapes used to conduct the label transfer experiment. We manually select 5 sources for each category characterizing various structures in the shape distribution.

AtlasNet. For AtlasNet, we train individual model for each category using two different settings: one using a sphere mesh as the template, dubbed Atlas-sph., and the other using 25 square meshes as the template, dubbed Atlas-25.

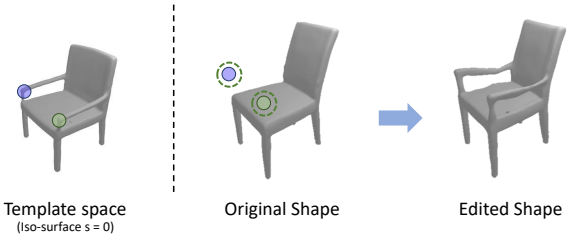

Figure III. Illustration of adding new structures to a given shape.

For each trained model, we first fit the template to all source shapes, and label each grid vertex on the template using label voting of its 5 nearest labeled points on 5 source shapes respectively. Then, given unlabeled points on a target shape, we fit the template to that shape, and label all points using label voting of 10 nearest grid vertices on the template.

SIF. We train SIF for each category using a template of 100 implicit kernels as in [23]. To conduct label transfer, we follow [23] to first calculate the template coordinates for all point on different shapes. The template coordinates has a dimension of 300 (100 kernels each with a 3-dimensional coordinates). Then, given an unlabeled point on the target shape, we find its 10 nearest labeled points in the template coordinate system and conduct label voting.

E. More Details of Shape Editing

Implementation Details. We achieve shape editing via Eq. (20) as described in the main paper. Given a shape with latent code α, we initialize the variable $\hat{\alpha}$ to α at the beginning of the optimization. Balancing weight for three terms in Eq. (20) are set to 1, 1, and 5 respectively. We use Adam optimizer with a learning rate of $1e-4$, and update $\hat{\alpha}$ with 1,000 iterations. The optimization process takes about 10 seconds.

Adding New Structures. To add new structures to a given shape, we select p_1' defined in Eq. (20) in the template space, as is shown in Fig. III. Then, we follow the same optimization process to learn $\hat{\alpha}$ for the new shape as described in the above paragraph.

More Editing Results. We show more shape editing results in Fig. VII. Our method is able to move selected points on the shape to desired position or add new structures. Features of the original shapes can be preserved after editing.

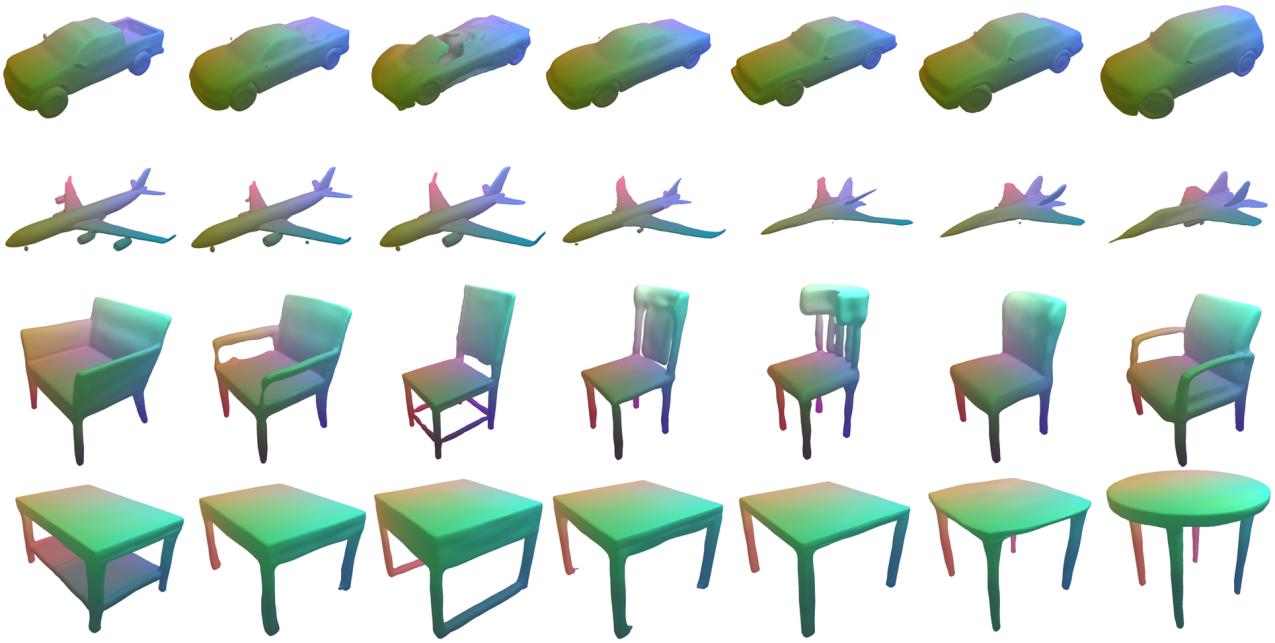

Figure IV. Latent code interpolation results. Interpolated shapes are in even columns.

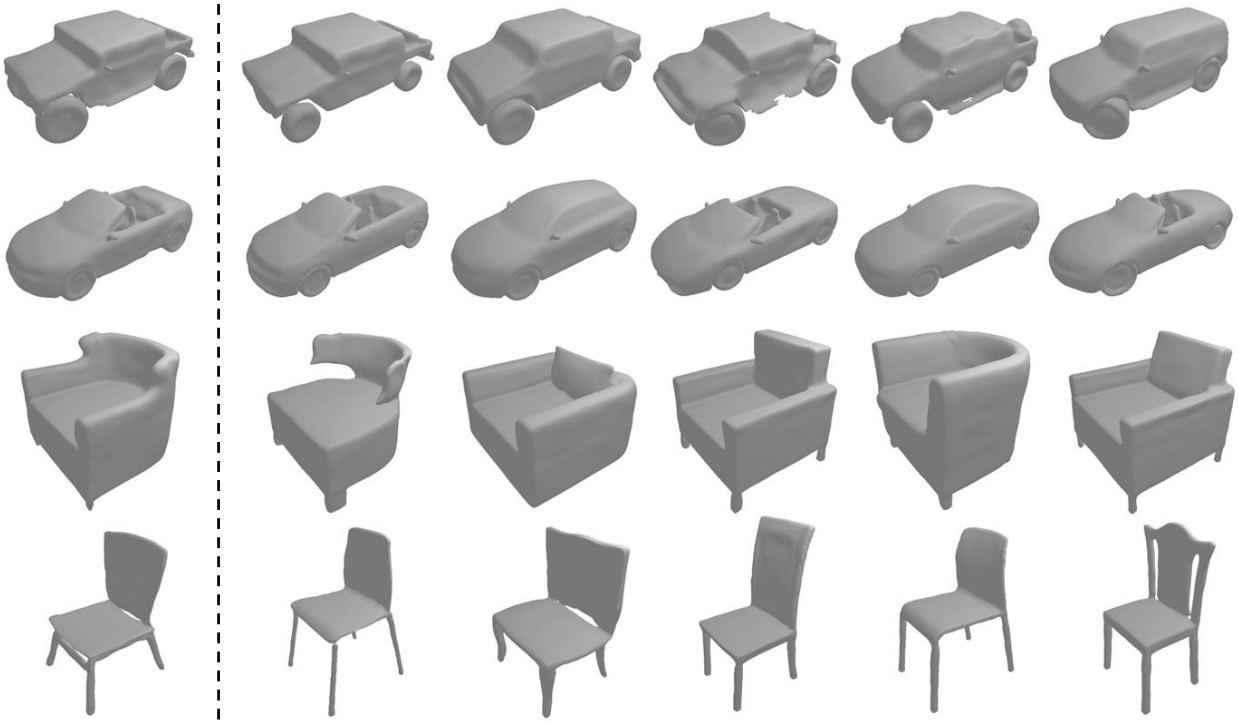

Figure V. Shape retrieval results using shape latent codes. Shapes on the right-hand side are nearest neighbors of the left-most ones and are sorted in an ascending distance order.

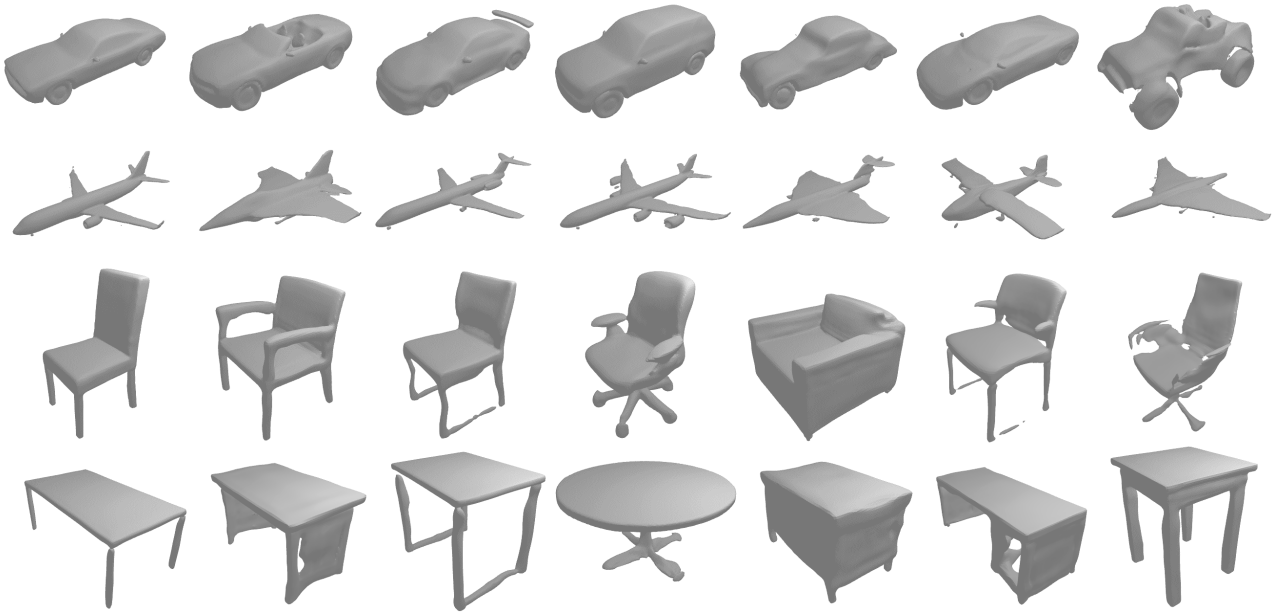

Figure VI. Sampled shapes from our model.

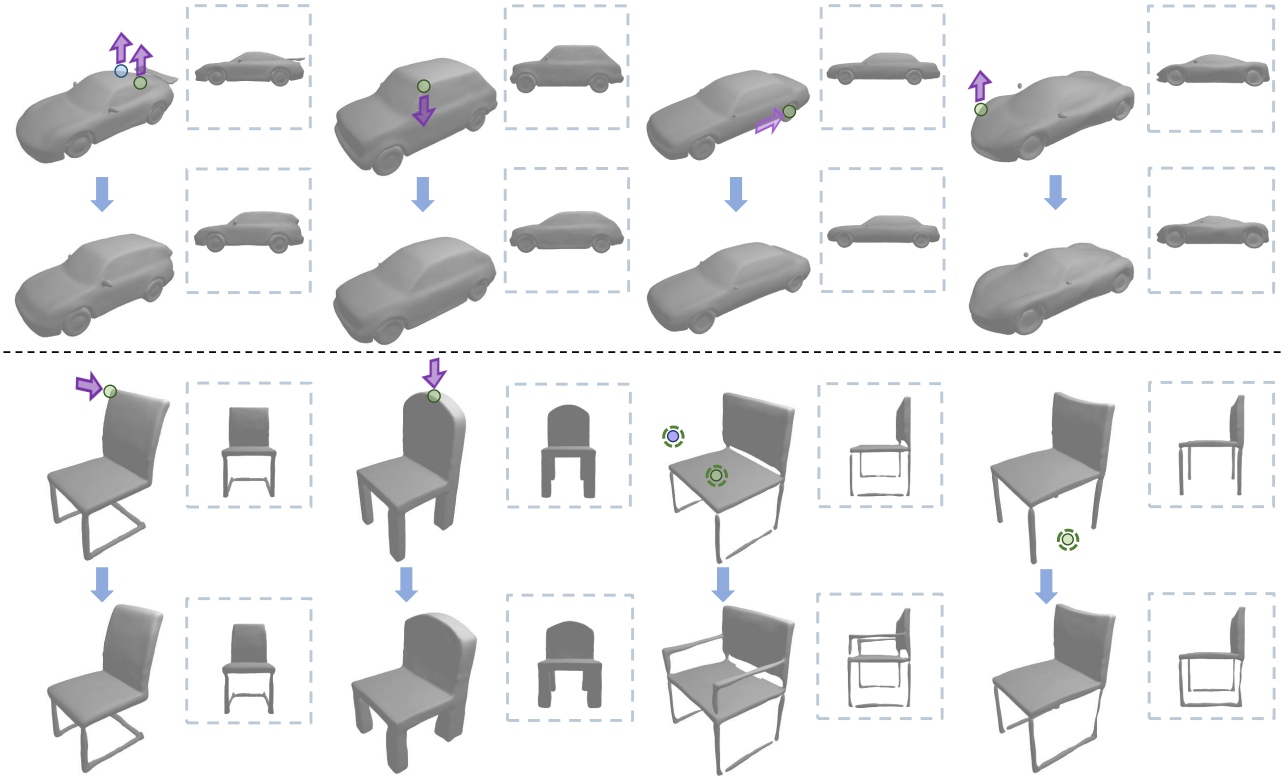

Figure VII. More shape editing results.