

DAE-Net: Deforming Auto-Encoder for fine-grained shape co-segmentation

Zhiqin Chen
Adobe Research
Seattle, USA
zchen@adobe.com

Hang Zhou
Simon Fraser University
Burnaby, Canada
hza162@sfsu.ca

Qimin Chen
Simon Fraser University
Burnaby, Canada
qca43@sfsu.ca

Hao Zhang
Simon Fraser University
Amazon
Burnaby, Canada
haoz@sfsu.ca

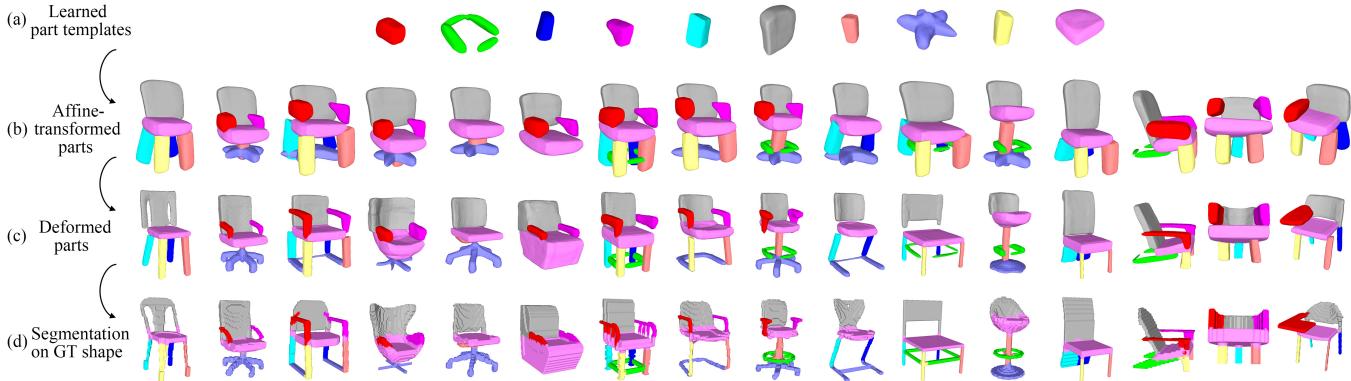


Figure 1: Our work DAE-Net co-segments a collection of 3D shapes into *fine-grained, consistent* parts. (a) The network learns a set of *part templates* shared by all shapes in the collection. (b) For each shape, DAE-Net selects the required parts and assembles them via affine transforms. (c) Each transformed part is further refined through constrained deformation. (d) The deformed parts are finally mapped to the input shapes to obtain the shape co-segmentation. Our network is trained with a shape reconstruction loss and several regularization losses, without any part supervision.

ABSTRACT

We present an unsupervised 3D shape co-segmentation method which learns a set of *deformable part templates* from a shape collection. To accommodate structural variations in the collection, our network composes each shape by a *selected subset* of template parts which are affine-transformed. To maximize the expressive power of the part templates, we introduce a per-part deformation network to enable the modeling of diverse parts with substantial geometry variations, while imposing constraints on the deformation capacity to ensure fidelity to the originally represented parts. We also propose a training scheme to effectively overcome local minima. Architecturally, our network is a *branched autoencoder*, with a CNN encoder taking a voxel shape as input and producing per-part transformation matrices, latent codes, and part existence scores, and the decoder outputting point occupancies to define the reconstruction loss. Our network, coined DAE-Net for Deforming Auto-Encoder,

can achieve unsupervised 3D shape co-segmentation that yields fine-grained, compact, and meaningful parts that are consistent across diverse shapes. We conduct extensive experiments on the ShapeNet Part dataset, DFAUST, and an animal subset of Objaverse to show superior performance over prior methods. Code and data are available at <https://github.com/czq142857/DAE-Net>.

CCS CONCEPTS

- Computing methodologies → Shape analysis.

KEYWORDS

Shape co-segmentation, analysis by synthesis, machine learning

1 INTRODUCTION

Co-analysis is a well-established paradigm for *unsupervised* learning over a data collection sharing some commonality, e.g., they all belong to the same category [Mitra et al. 2013; Xu et al. 2016]. The central problem of co-analyzing a 3D shape collection is *shape*

©Authors | ACM 2024. This is the author’s version of the work. It is posted here for your personal use. Not for redistribution. The definitive version of record was published in SIGGRAPH Conference Papers ’24, <https://doi.org/10.1145/3641519.3657528>.

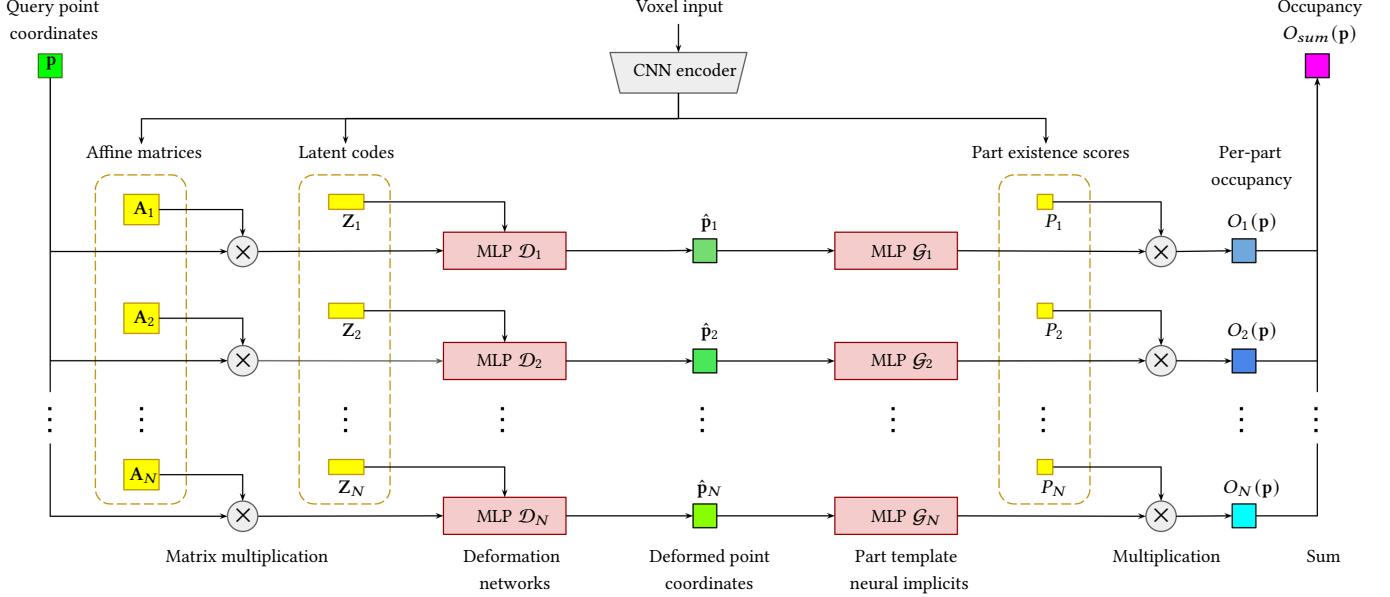


Figure 2: Network architecture of DAE-Net. Our network consists of N branches representing N parts of a 3D shape. To reconstruct part i , the query point coordinates in world frame are first transformed into the local frame of the part using an affine matrix that is predicted by a shape encoder network, a CNN. The transformed local coordinates are further deformed by a deformation MLP \mathcal{D}_i conditioned on a latent code, that is both shape- and part-specific and also produced by the CNN, to refine the part details. Finally, the deformed local coordinates are fed into a part template MLP \mathcal{G}_i to produce the occupancy of the query point. The occupancy is multiplied by the predicted part existence score from the CNN, so that the occupancy is set to zero if the part does not exist in the shape. We sum the occupancies from all N parts to obtain the occupancy of the query point on the entire shape, which is used to compute the reconstruction loss.

co-segmentation¹, whose goal is to learn a consistent segmentation of all the shapes in the collection [Chen et al. 2019; Golovinskiy and Funkhouser 2009a; Huang and Guibas 2013; Paschalidou et al. 2019; Sidi et al. 2011; Tulsiani et al. 2017; Yang et al. 2021; Zhu et al. 2020]. On top of a structural understanding per shape, a co-segmentation induces part correspondences across the collection to facilitate a variety of downstream tasks including attribute and knowledge transfer, shape editing or co-editing with part control, and generative modeling of shape structures [Chaudhuri et al. 2020].

The most difficult challenge to shape co-segmentation arises when there are significant structural *and* geometric variations across the shape collection; see last row of Figure 1. Insisting on a single template for part organization, even a hierarchical one [van Kaick et al. 2013], would impose too much structural rigidity. As a result, the learned template is necessarily coarse to fulfill the consistency requirement across the whole collection. On the flip side, abstraction-based methods using predefined primitives such as cuboids or superquadrics place rigidity on the modeling of part geometries, often resulting in over-segmentation and less meaningful correspondences between the fragmented parts.

In this work, we present an unsupervised shape co-segmentation method which learns a set of *deformable part templates* from a

collection of 3D shapes belonging to the same category. As shown in Figure 1, each learned part template (top row) models a set of corresponding parts in the shape collection. Our network is trained to transform, via affine transformations, and then deform a selected *subset* of the template parts to best reconstruct each shape in the collection, without any part annotations as supervision.

The overall architecture of our network, as depicted in Figure 2, is that of an N -branch autoencoder, with N being an upper bound on the number of template parts for the shape category. A CNN encoder takes a voxelized shape as input and produces the affine matrices, part latent codes (to condition the deforming MLPs), and part existence scores as a means to select among the part template MLPs to reconstruct the input shape. The part templates are modeled as neural implicit functions, with the decoder outputting point occupancies to define the reconstruction loss.

Our network is coined *DAE-Net* for *Deforming Auto-Encoders*. The key idea behind our approach arises from the stipulation that corresponding parts in different shapes should have approximately the same shape (or form). This is inspired by the well-known design principle, “form follows function,” while part correspondence is ultimately about functional correspondence. On the technical front, the design of our template learning has been inspired by Transforming Auto-encoders (TAE) [Hinton et al. 2011] for learning the first level of Capsule networks. Like TAE, our work represents a 3D shape via affine-transformed parts selected from a group of learned part templates, to accommodate structural variations and produce

¹It is worth distinguishing 3D shape co-segmentation from object co-segmentation over an image collection, e.g., [Chen et al. 2012; Vicente et al. 2011]. The latter is a special case of image segmentation with the goal of jointly segmenting or *detecting* semantically similar *objects* out of multiple images or video frames.

a fine-grained co-segmentation. However, the limited expressive power of affine transformations is not sufficient for faithfully representing parts with substantial geometric variations. Our per-part deformation alleviates such part modeling rigidity to learn diverse parts, while imposing constraints on the deformation capacity to ensure fidelity to the originally represented parts.

In addition, we propose a training scheme to effectively overcome *local minima* encountered during training. While BAE-Net, RIM-Net, and our method are all built on the branched auto-encoder structure and may suffer from local minima, only our method can adopt the proposed training scheme to overcome this issue. This is because the training scheme includes re-initializing certain auto-encoder branches, which is trivial in our method as it uses different networks to represent different parts (branches). However, it is non-trivial for the other two works since they produce all branches by the same MLP or require complex hierarchical training.

Through extensive experiments, we show that DAE-Net can achieve unsupervised 3D shape co-segmentation that yields fine-grained, compact, and meaningful parts that are consistent across diverse shapes. With comparisons conducted on the Shapenet Part dataset [Chang et al. 2015; Yi et al. 2016], DFAUST [Bogo et al. 2017], and an animal subset of Objaverse [Deitke et al. 2023], DAE-Net exhibits superior performance over prior unsupervised shape segmentation methods. Additionally, we demonstrate shape clustering using the part existence score produced by our method and showcase a controllable shape detailization application enabled by our segmentation results.

2 RELATED WORK

3D co-segmentation with handcrafted priors. Early works in geometry processing utilize fundamental geometric cues [Shamir 2008], such as surface area, curvature, and geodesic distance, to derive higher-level semantic attributes for 3D shape segmentation. Golovinskiy and Funkhouser [Golovinskiy and Funkhouser 2009b] investigated consistent co-segmentation of 3D shapes by constructing a graph connecting corresponding faces across different meshes. The unsupervised clustering approach was later extended to feature spaces via diffusion maps and spectral clustering [Sidi et al. 2011]. Thereafter, extensive research has been devoted to co-analysis of sets of shapes based on various clustering strategies [Hu et al. 2012; Huang and Guibas 2013; Huang et al. 2011; Meng et al. 2013; Xu et al. 2010]. [Shu et al. 2016] adapted the setting by transforming the handcrafted local features with stacked auto-encoders before per-shape graph cuts. In contrast, our approach is an end-to-end differentiable pipeline and free of externally introduced design elements previously proposed for shape segmentation.

Co-segmentation via 3D shape reconstruction. To circumvent 3D annotations, weakly-supervised and unsupervised learning schemes utilize large collections of unlabelled data for segmentation. [Tulsiani et al. 2017] introduced sets of cuboids to approximate a 3D shape without supervision. [Sun et al. 2019] and [Yang and Chen 2021] improved the cuboid representation to model complex shape structures. [Paschalidou et al. 2019] utilized superquadrics for shape abstractions, leading to better shape parsing ability. [Deprelle et al.

2019] represented shapes as deformation and combination of elementary structures represented in point clouds for shape reconstruction and unsupervised correspondence. BAE-Net [Chen et al. 2019] proposed a branched auto-encoder network for shape co-segmentation, where each individual branch learns to localize part instances across multiple samples. BSP-Net [Chen et al. 2020] and CvxNet [Deng et al. 2020] represented shapes as convex polytopes using neural implicit fields, allowing shape co-segmentation in the view of commonly associated convexes. [Paschalidou et al. 2020] built a binary tree of primitives for shape reconstruction without part-level supervision. [Kawana et al. 2020] represented shapes as star primitives, where each primitive is formed by a continuous function defined on the sphere surface. Neural Parts [Paschalidou et al. 2021] defined primitives as invertible neural networks, allowing inverse mapping between a sphere and the target part for efficient computation. RIM-Net [Niu et al. 2022] represented shapes as hierarchical shape structures with recursive implicit fields. Part-NeRF [Tertikas et al. 2023] represented objects as a collection of locally defined Neural Radiance Fields (NeRFs), and designed a part-aware generative model based on auto-encoders. DPF-Net [Shuai et al. 2023] performed structured shape reconstruction by representing parts as deformed cuboids and cylinders. [Huang et al. 2023] proposed to reconstruct part primitives from multi-view images while imposing convexity regularization on the parts. Several works [Deng et al. 2021; Kim et al. 2023; Zheng et al. 2021] coupled neural implicit fields with neural deformation fields to learn dense correspondences between shapes. Other than implicit representations, continuous progress of co-segmentation has been made for point clouds [Yang et al. 2022; Zhu et al. 2020].

Our work learns the *shapes* of a set of part templates. In contrast, prior works either had no explicit guidance for defining a part (e.g., BAE-Net, RIM-Net), merely relying on MLPs to perform segmentation, or make much stronger assumptions about the part templates, e.g., as cuboids, convexes, or other basic primitives. Our template represents the “mean” shape of the parts, and we use a subset of those templates to build each 3D shape via per-template affine transformation and local deformation. The learned templates also help achieve compactness of the segmentation, which is hard for primitive templates such as cuboids.

Zero-shot 3D segmentation using pretrained models. Zero-shot segmentation aims to make predictions for categories that are not annotated in training. [Michele et al. 2021] extended zero-shot semantic image segmentation [Bucher et al. 2019] to 3D, which is followed by [Chen et al. 2022; Koo et al. 2022]. With the emergence of NeRFs [Lombardi et al. 2019; Mildenhall et al. 2021], methods have been developed to model semantic fields [Fan et al. 2022; Fu et al. 2022; Hong et al. 2023; Kundu et al. 2022; Siddiqui et al. 2023; Tschernezki et al. 2022; Vora et al. 2021; Zhi et al. 2021] by reconstructing semantic annotations from multi-view renderings.

Going beyond zero-shot learning, the more general open-vocabulary setting [Ding et al. 2023] assumes a large vocabulary corpus is accessible during training. Recently, great strides have been made in vision-language pretraining [Alayrac et al. 2022; Jia et al. 2021; Saharia et al. 2022; Zhang et al. 2022] by pretraining large-scale image-text pairs. Owing to learned rich visual concepts and notable zero-shot capabilities, 3D Highlighter [Decatur et al. 2023]

and SATR [Abdelreheem et al. 2023b] proposed mesh segmentation based on off-the-shelf CLIP [Radford et al. 2021] and GLIP [Li et al. 2022b] models, and PartSLIP [Liu et al. 2023] relied on GLIP for point cloud segmentation. [Abdelreheem et al. 2023a] utilized large foundation vision and language models to perform co-segmentation between pairs of shapes. By distilling zero-shot image segmentation models [Caron et al. 2021; Li et al. 2022a] into NeRFs, [Goel et al. 2023; Kobayashi et al. 2022; Peng et al. 2023] are able to perform scene segmentation under open-vocabulary settings. In contrast, our method does not rely on any pretrained image-language models and is flexible on its own.

Transforming Auto-encoders. [Hinton et al. 2011] introduced Transforming Auto-encoders (TAE) to learn the first level of Capsule networks [Sabour et al. 2017]. TAE learns to recognize visual entities, i.e., parts, and their poses, in an unsupervised manner by training an auto-encoder on a set of images. Specifically, each part i is represented by a distinct capsule, which consists of a recognition module and a generation module. The recognition module predicts the probability P_i that the part is present in input image \mathcal{I} , as well as the transformation of the part with respect to its canonical pose, e.g., translation (x_i, y_i) . The generation module \mathcal{G}_i represents the “shape” of the part; it inputs the transformed coordinates and outputs the transformed part. The reconstructed image is obtained by

$$\mathcal{I}_{rec} = \sum_i P_i \cdot \mathcal{G}_i(x_i, y_i), \quad (1)$$

where the parts are selected by P_i and transformed by (x_i, y_i) to assemble the final output image.

3 METHOD

In this section, we introduce the network architecture and loss functions of our method. We also present a training scheme to prevent the model from being stuck in a local minimum during optimization. Code and data are also provided in the [Supplementary](#).

3.1 Network architecture

The network architecture of DAE-Net is shown in Figure 2. In our network, we use N branches to present N parts of the shape \mathcal{V} to be reconstructed, where each part i has a dedicated MLP \mathcal{G}_i to represent the corresponding part template as a neural implicit [Chen and Zhang 2019; Joon Park et al. 2019; Mescheder et al. 2019], and a dedicated deformation MLP \mathcal{D}_i for part deformation. The shape encoder \mathcal{E} is a 3D CNN that takes an occupancy voxel grid $\mathcal{V} \in \{0, 1\}^{64 \times 64 \times 64}$ as input and produces per-part affine transformation matrices $\mathbf{A}_i^{\mathcal{V}} \in \mathbb{R}^{3 \times 4}$, latent codes $\mathbf{Z}_i^{\mathcal{V}} \in \mathbb{R}^4$, and part existence scores $P_i^{\mathcal{V}} \in [0, 1]$, for part i .

In the following, we assume that the outputs are all from shape \mathcal{V} and therefore drop the superscript \mathcal{V} for simplicity. We also require densely sampled points from the shape \mathcal{V} to train the neural implicit, and we denote the occupancy of a sampled point $\mathbf{p} \in \mathbb{R}^3$ as $\mathcal{V}[\mathbf{p}] \in \{0, 1\}$.

To reconstruct part i , the query point \mathbf{p} in world coordinates is first projected to homogeneous coordinates $\mathbf{p}' \in \mathbb{R}^4$, i.e., appending 1 to \mathbf{p} , and then transformed into the local frame of the part via an

affine transformation: $\mathbf{p}_i^{local} = \mathbf{A}_i \mathbf{p}'$. The local coordinates \mathbf{p}_i^{local} are further deformed by the deformation MLP \mathcal{D}_i conditioned on the latent code \mathbf{Z}_i . Note that \mathcal{D}_i predicts the offsets of the deformation: $\Delta \mathbf{p}_i^{local} = \mathcal{D}_i(\mathbf{p}_i^{local}, \mathbf{Z}_i)$, therefore the deformed coordinates are $\hat{\mathbf{p}}_i = \mathbf{p}_i^{local} + \Delta \mathbf{p}_i^{local}$.

Finally, the deformed coordinates $\hat{\mathbf{p}}_i$ are fed into the part template MLP \mathcal{G}_i to produce the occupancy $\mathcal{G}_i(\hat{\mathbf{p}}_i) \in [0, 1]$. The occupancy is multiplied by the predicted part existence score P_i , so that the occupancy will be set to zero if the part is not deemed to exist in the shape. The final occupancy for point \mathbf{p} on part i is $O_i(\mathbf{p}) = P_i \cdot \mathcal{G}_i(\hat{\mathbf{p}}_i)$.

To obtain occupancy of the query point on the entire shape, we sum the occupancies from all parts: $O_{sum}(\mathbf{p}) = \sum_i O_i(\mathbf{p})$. Since the parts should ideally be non-overlapping, i.e., the per-part occupancies should be one-hot, $\sum_i O_i(\mathbf{p}) = \max_i O_i(\mathbf{p}) = 1$, for occupied points, we also use an auxiliary shape occupancy $O_{max}(\mathbf{p}) = \max_i O_i(\mathbf{p})$ to compute the reconstruction losses.

3.2 Loss functions

First, we use a shape reconstruction loss,

$$\mathcal{L}_{recon}^{sum} = \mathbb{E}_{\mathbf{p}}(O_{sum}(\mathbf{p}) - \mathcal{V}[\mathbf{p}])^2. \quad (2)$$

to supervise the entire model. As mentioned, to encourage non-overlapping parts, we use a secondary reconstruction loss,

$$\mathcal{L}_{recon}^{max} = \mathbb{E}_{\mathbf{p}}(O_{max}(\mathbf{p}) - \mathcal{V}[\mathbf{p}])^2. \quad (3)$$

Second, we need to constrain the per-part deformation, so that the deformation only changes the part locally and does not transform it into a different part or multiple parts. We use a simple L_2 loss on the predicted deformation offsets to achieve this

$$\mathcal{L}_{deform} = \mathbb{E}_{\mathbf{p}} \mathbb{E}_i \|\Delta \mathbf{p}_i^{local}\|_2^2. \quad (4)$$

Finally, we have a loss to control the sparsity of the segmented parts, which is achieved by penalizing the predicted part existence scores. Specifically, in each training iteration, for a mini-batch of shapes \mathcal{S} , we have

$$\mathcal{L}_{sparse} = -\mathbb{E}_i(1 - \mathbb{E}_{\mathcal{V} \in \mathcal{S}} P_i^{\mathcal{V}})^2, \quad (5)$$

where $\mathbb{E}_{\mathcal{V} \in \mathcal{S}} P_i^{\mathcal{V}}$ evaluates what fraction of the shapes in \mathcal{S} contains part i . If part i is rare, $\mathbb{E}_{\mathcal{V} \in \mathcal{S}} P_i^{\mathcal{V}}$ is close to zero and P_i will receive a greater penalty, therefore effectively making uncommon parts disappear.

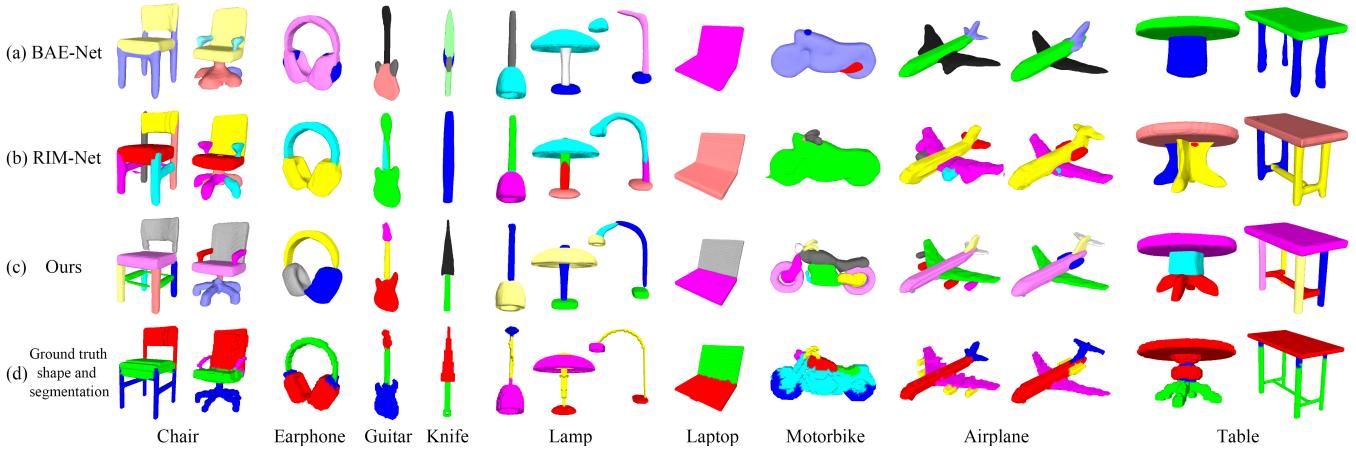
The overall loss is a weighted sum of the loss terms,

$$\mathcal{L} = \mathcal{L}_{recon}^{sum} + \alpha \mathcal{L}_{recon}^{max} + \beta \mathcal{L}_{deform} + \gamma \mathcal{L}_{sparse}. \quad (6)$$

We set $\alpha = 0.1$ and $\beta = 100$. γ needs to be set according to the desired granularity of the segmentation; see Figure 6. Typically, different shape categories require different γ values to achieve desirable segmentation quality, therefore tuning γ is often necessary. In our experiments, we try $\gamma = 0.02, 0.01, 0.002$, and 0.001 in decreasing order until reaching the desired granularity.

Table 1: Quantitative results on shape segmentation compared to BAE-Net [Chen et al. 2019] and RIM-Net [Niu et al. 2022], evaluated by average per-part IOU. Best results are marked in bold.

	Mean	plane	bag	cap	chair	earph.	guitar	knife	lamp	laptop	motor.	mug	pistol	rocket	skateb.	table
BAE-Net	56.2	59.8	84.4	84.9	54.1	44.1	51.0	32.5	74.7	27.1	27.5	94.4	29.0	40.9	63.3	75.7
RIM-Net	53.6	52.7	86.1	62.6	79.2	72.9	25.7	29.5	68.3	33.2	28.5	48.6	36.2	39.5	64.9	76.0
Ours	76.9	78.0	84.4	86.3	85.5	77.2	88.4	85.8	73.2	95.0	48.1	94.2	74.6	38.7	68.2	75.5

**Figure 3: Qualitative results on shape segmentation compared to BAE-Net [Chen et al. 2019] and RIM-Net [Niu et al. 2022] on ShapeNet Part dataset [Chang et al. 2015; Yi et al. 2016]. Within the same category, same color indicates the parts are from the same branch of the network, thus are considered to be corresponded. Since the ground truth segmentation in ShapeNet Part dataset is on point clouds, we color the voxels in (d) using nearest neighbor.**

3.3 Overcoming local minima

Since unsupervised co-segmentation via reconstruction is an ill-posed problem, the training is often stuck in a local minimum, where a predicted part represents multiple ground truth parts, or a number of predicted parts represent a single ground truth part, and further training cannot break the part apart or merge the parts together; see Figure 5 (a).

To devise a training scheme to overcome local minima, we partition the training into K eras, each containing M iterations. Then we design an operation $\text{revive}(i)$, to re-initialize the weights of \mathcal{D}_i , \mathcal{G}_i , and the portion of the last layer of \mathcal{E} that affects A_i , Z_i , and P_i . In the first era, we initialize \mathcal{E} and revive all the branches. We explicitly track the age of each branch, i.e., how many eras a specific branch has lived after its last revival. At the end of each era, we compute $\mathbb{E}_{V \in S} P_i^V$ for all the training shapes S , which indicates what fraction of the training shapes contains part i . If the fraction for part i is lower than a threshold (10%), the branch is considered “dead” and will be revived. We also revive the oldest branch, so the part it represents has the option of breaking down into smaller parts or being merged into other parts. Reviving the oldest branch holds the risk that it can make the model stuck in a worse local minimum. Therefore, at the end of each era, we also compute the average reconstruction error evaluated by IoU (Intersection over Union) of per-point occupancy, and compare it with the IoU of the previous era. If the current IoU is better than the previous one, we keep the current network weights; otherwise we load the network

weights of the previous era. This is based on the assumption that a better segmentation should always lower the reconstruction error.

3.4 Training details

In the experiments, we train individual models for different object categories. We set branch number $N = 16$. However, for categories with evidently less parts, such as laptops and mugs, we set $N = 8$ to save training time. We train each model $2N$ eras with each era consisting of 125,000 iterations of training. In each iteration, we set the mini-batch size to 16. The model is trained with Adam optimizer [Kingma and Ba 2015] with a learning rate of 0.0002. Training on one category takes 8 hours when $N = 16$ and 2.5 hours when $N = 8$, on one NVIDIA V100 GPU.

4 EXPERIMENTS

Our experiments mainly focus on unsupervised co-segmentation. Since our method predicts an existence score for each shape part, we can group the shapes in the dataset according to which parts they have. Finally, we show that our segmentation results can be easily combined with DECOR-GAN [Chen et al. 2021] to achieve shape detailization while controlling per-part geometric style.

4.1 Unsupervised shape co-segmentation

We perform experiments on shapes from 15 categories in the ShapeNet Part dataset [Chang et al. 2015; Yi et al. 2016]; the car category

is excluded since its ground truth segmentation consists of non-volumetric parts; see Section 5 and Figure 7 (a) for details. We compare with BAE-Net [Chen et al. 2019] and RIM-Net [Niu et al. 2022], which also perform unsupervised shape co-segmentation. There are other works that perform shape abstraction thus can potentially be used for shape segmentation, but they either over-segment the shape into excessive numbers of parts [Chen et al. 2020; Deng et al. 2020], or do not show consistent part correspondence across shapes [Deprelle et al. 2019; Paschalidou et al. 2021, 2019], therefore they are not being compared in the experiments. For all methods, we train an individual model on all shapes in each shape category. All methods are trained on pre-processed voxel data provided by [Chen et al. 2019]. We use the default settings of BAE-Net and RIM-Net to train their models, except that their original branch numbers are 8, therefore we try both 8 and 16 as their branch numbers for each category and show the best results.

We quantitatively evaluate the co-segmentation accuracy by per-part IOU averaged on all parts and all shapes in the category, which is the standard evaluation metric used for shape segmentation on the ShapeNet Part dataset. To infer segmentation from the network, for each query point p , its label will be assigned as the label of the network branch that produces the maximum occupancy, i.e., $\text{label}(p) = \text{label}(\arg \max_i O_i(p))$. To achieve objectiveness and fairness in evaluation, for all methods, we use an algorithm to automatically label each output branch of the networks with a semantic part label so that the average IOU is maximized. The automatic labeling algorithm is implemented by exhaustive search. The average IOU is computed on the standard testing split of the ShapeNet Part dataset for each category, so that the reported IOU can be directly compared with results from other semi-supervised or fully-supervised methods on this dataset.

As shown by the quantitative results from Table 1, our method outperforms the compared methods by a large margin on most categories. On the few categories where the other methods performed better, our method is often less than 2% worse than the best-performing alternative.

Importantly however, these quantitative results do not tell the whole story, since the ground truth segmentation is coarse, e.g., chair is only segmented into 4 high-level parts: back, seat, leg, and arm. DAE-Net has been designed to provide *fine-grained* segmentation, as shown in Figure 3, while other methods either fail to segment finer parts or have messy segmentations. We also show the learned part templates in the *Supplementary*.

In addition, we trained all methods on two additional datasets: DFAUST [Bogo et al. 2017] that contains human body shapes, and a subset of Objaverse [Deitke et al. 2023] that contains quadruped animals. Qualitative results are shown in Figure 4 and Figure 11, where our method has clearly better performance. Moreover, since our method produces clean segmentation with part correspondences between shapes, we can easily build a skeleton of the shape using our segmentation results, as shown in Figure 4 (e) and Figure 11 (e). Since DFAUST and Objaverse do not have ground truth segmentation, quantitative results are hard to obtain. Therefore, we provide more details and ample qualitative results in the *Supplementary*.

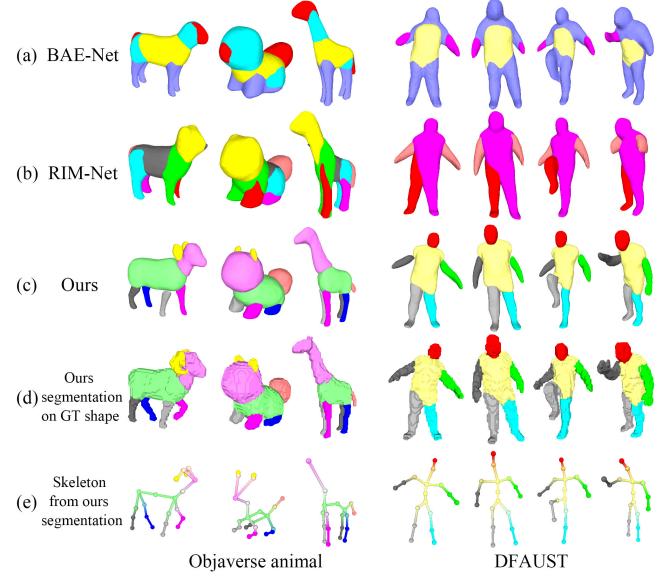


Figure 4: Qualitative results on shape segmentation on an animal subset of Objaverse [Deitke et al. 2023], and DFAUST [Bogo et al. 2017]. We also show the skeletons built upon our segmentation. More results can be found in Figure 11 and the *Supplementary*.

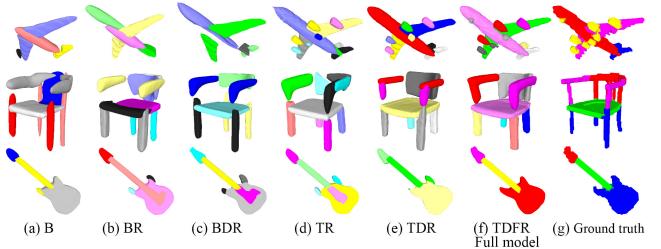


Figure 5: Qualitative results of Ablation study on airplane, chair, and guitar. See Section 4.2 for the meaning of the abbreviations.

4.2 Ablation studies

Our full model is made of several components. We start from the base model (**B**) where we have N MLP branches \mathcal{G}_i , each taking point coordinates p and shape latent code as input and outputting the occupancy of p in part i ; the per-part occupancies are weighted by the predicted part existence scores P_i to produce the final output. In the base model **B**, we use both $\mathcal{L}_{\text{recon}}^{\text{sum}}$ and $\mathcal{L}_{\text{recon}}^{\text{max}}$ in the loss function. We could use only $\mathcal{L}_{\text{recon}}^{\text{sum}}$ for training, denoted as **B_S**; or only $\mathcal{L}_{\text{recon}}^{\text{max}}$, denoted as **B_M**. Next, we introduce transforming auto-encoder, denoted as **T**, where \mathcal{G}_i is no longer conditioned on latent codes since it is now a part template shared by all shapes; and the templates will be affine-transformed by the predicted matrices. For both **B** and **T**, we further add the deformation networks \mathcal{D}_i to deform each individual part, denoted as **D**. We include $\mathcal{L}_{\text{deform}}$

Table 2: Ablation study on shape segmentation, evaluated by average per-part IOU. See Section 4.2 for the meaning of the abbreviations. Mean is the mean IOU on all 15 categories. We also report some IOUs on representative categories.

	B _S	B _M	B	BD	BDF	T	TD	TDF	B _S R	B _M R	BR	BDR	BDFR	TR	TDR	TDFR (Full model)
Mean	59.1	45.1	58.6	62.3	57.8	61.5	68.9	68.6	69.3	60.6	73.2	72.6	70.3	74.6	74.5	76.9
Plane	56.2	30.5	53.3	71.2	62.6	60.8	65.7	73.6	71.8	53.8	71.3	74.4	72.5	75.1	78.0	78.0
Chair	73.3	49.2	60.5	69.9	73.4	69.3	74.9	76.0	84.7	57.6	84.5	83.5	84.0	85.2	84.2	85.5
Guitar	52.9	27.5	78.4	46.4	53.6	70.2	84.0	73.2	77.1	37.1	85.5	86.6	86.4	88.1	84.1	88.4

Table 3: Quantitative results on few-shot shape classification, and number of shapes for each category in ShapeNet Part dataset.

	plane	bag	cap	chair	earph.	guitar	knife	lamp	laptop	motor.	mug	pistol	rocket	skateb.	table
Classification precision	0.93	0.79	N/A	0.97	N/A	0.96	0.83	0.86	0.96	0.84	0.76	0.87	0.44	0.30	0.91
Classification recall	0.98	0.14	0.00	0.96	0.00	0.99	0.94	0.73	1.00	0.86	0.90	0.85	0.73	0.07	0.91
Number of shapes	2,690	76	55	3,746	69	787	392	1,546	445	202	184	275	66	152	5,263

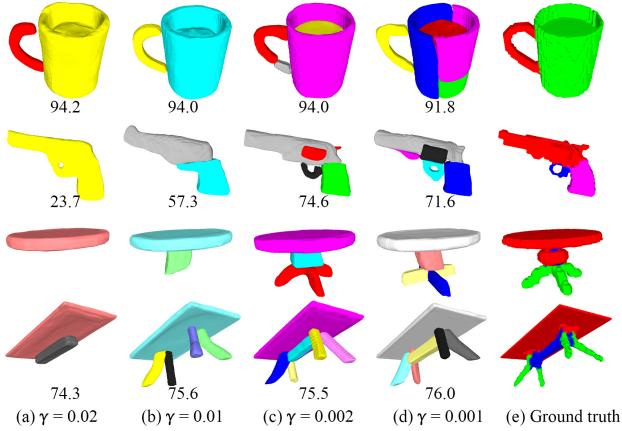


Figure 6: Ablation study on the weight γ of the sparsity loss $\mathcal{L}_{\text{sparse}}$ on mug, pistol, and table. The number under each shape shows the IOU of its category when trained with a specific γ value.

Note that the sparsity loss $\mathcal{L}_{\text{sparse}}$ is applied to all the cases above. Its weight γ in the final objective function controls the granularity of the segmentation results. We vary γ for some shape categories and show quantitative and qualitative results in Figure 6.

4.3 Shape clustering

After thresholding the predicted part existence scores P_i with a pre-defined threshold (0.5), we obtain a binary N -d vector representing whether part i exists in the output shape or not. We can then group the shapes into sub-categories according to this vector. In Figure 8, we show some groups for airplane and chair. The clustering produces groups that contain structurally similar shapes.

We additionally train a single model on all categories in the ShapeNet Part dataset, which produces 68 groups that contain no less than 10 shapes. We label each group with the closest category label, as shown in Figure 9. For other groups with less than 10 shapes, we assign to them an “N/A” label. Therefore, we have effectively classified all the shapes into different categories. We compare our classification results with the ground truth to obtain the quantitative results provided in Table 3. Our “few-shot” classification model has achieved decent results on categories with abundant shapes but ignored rare categories.

4.4 Application: part-level shape detailization

DECOR-GAN [Chen et al. 2021] is a method for conditional voxel upsampling. When trained with M detailed shapes representing M styles, the user can upsample an input coarse voxel grid into a high-resolution, detailed voxel model, but with only one of the M styles learned from the training shapes. The lack of part-level control makes it impossible to combine styles from different shapes, as shown in Figure 10 (a).

With our predicted segmentation, we provide DECOR-GAN the ability to control per-part geometric styles. In the original DECOR-GAN, each input voxel is associated with one of the M styles. But during training, all voxels from the input shape have to be associated with the same style because there is no part-level segmentation available. We simply modify the training procedure to assign voxels from different parts with different styles. We test our approach on

to constrain the deformation, denoted as \mathbf{F} . Finally, we apply our training scheme to overcome the local minima, denoted as \mathbf{R} .

Table 2 summarizes the quantitative results, where we report the mean IOU on all categories and some IOUs on representative categories. We also show some qualitative results in Figure 5. Our training scheme R plays a major role in improving the results. Methods trained without R tend to be stuck in local minima and cannot be improved further. B_M and B_{MR} perform much worse than B_S and B_{SR} , showing that $\mathcal{L}_{\text{recon}}^{\text{max}}$ is not as easy to optimize as $\mathcal{L}_{\text{recon}}^{\text{sum}}$ since its gradient cannot be propagated to all network branches. However, $\mathcal{L}_{\text{recon}}^{\text{max}}$ can help some categories achieve better results, as shown by B_{SR} vs. BR . Transforming auto-encoder with our training scheme (TR) can already achieve impressive results, but after augmented with deforming networks (D) with constraints (F), our full model (TDFR) performs the best.

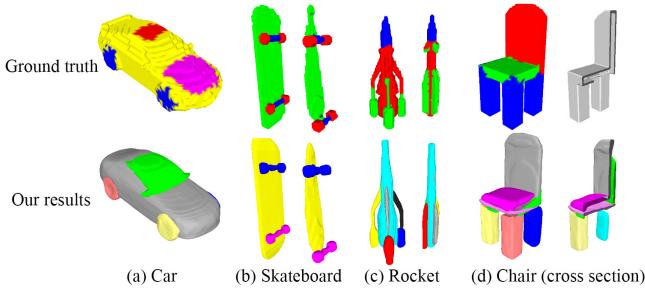


Figure 7: Unsuccessful segmentation results.

two categories: one is a combined chair+table category, and the other is the plant category. We use the data provided by DECOR-GAN, which are originally from ShapeNet [Chang et al. 2015]. Some qualitative results are shown in Figure 10.

5 CONCLUSIONS

We have introduced the Deforming Auto-Encoder, or DAE-Net, which extends the idea in Transforming Auto-encoders for unsupervised part learning. With our part-wise deformation networks improving the shape reconstruction quality and our training scheme effectively overcoming the local minima issue, our work is the first to show that the TAE framework can yield high-quality, consistent, and fine-grained 3D shape co-segmentation.

Our method segments a shape by reconstructing and deforming individual parts of the shape. As we adopt a volumetric shape representation, our method is unable to segment surface parts, e.g., hood and roof in the car category of ShapeNet; see Figure 7 (a). Likewise, our method may not segment certain semantic parts, as it does not have actual semantic understanding, e.g., the connecting part of the earphone category (blue in ground truth) in Figure 3, and the wheels (red) and head (blue) of the skateboard and rocket categories in Figures 7 (b) and (c), respectively.

Our method also relies heavily on the quality of the training data. Although our training data has been pre-processed to make the voxels as solid as possible, there are still shapes that are hollow inside, e.g., see Figure 7 (d). These hollow shapes will not be correctly segmented, and they may even negatively affect the segmentation quality of other shapes. On a related matter, parts will not be correctly segmented if they are not correctly reconstructed by our method, either due to underfitting or the parts being too small, as shown in the second rocket example in Figure 7 (c).

In the future, we could seek better granularity control by introducing hierarchical structures in the shape representation, as in RIM-Net [Niu et al. 2022]. Combining our method with open-vocabulary semantic segmentation methods is also an interesting future direction, which may help produce consistent and meaningful cross-category co-segmentation.

ACKNOWLEDGMENTS

We thank all the anonymous reviewers for their insightful comments and constructive feedback. This work was supported in part by an NSERC Discovery Grant (No. 611370) and Adobe gift funds.

REFERENCES

- Ahmed Abdelreheem, Abdelrahman Eldekokey, Maks Ovsjanikov, and Peter Wonka. 2023a. Zero-shot 3D shape correspondence. In *SIGGRAPH Asia 2023 Conference Papers*. 1–11.
- Ahmed Abdelreheem, Ivan Skorokhodov, Maks Ovsjanikov, and Peter Wonka. 2023b. SATR: Zero-shot semantic segmentation of 3D Shapes. In *ICCV*.
- Jean-Baptiste Alayrac et al. 2022. Flamingo: a visual language model for few-shot learning. *NeurIPS* (2022).
- Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2017. Dynamic FAUST: Registering Human Bodies in Motion. In *CVPR*.
- Maxime Bucher, Tuan-Hung Vu, Matthieu Cord, and Patrick Pérez. 2019. Zero-shot semantic segmentation. *NeurIPS* (2019).
- Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging properties in self-supervised vision transformers. In *CVPR*.
- Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015. ShapeNet: An Information-Rich 3D Model Repository. *arXiv* (2015).
- Siddhartha Chaudhuri, Daniel Ritchie, Jiajun Wu, Kai Xu, , and Hao Zhang. 2020. Learning Generative Models of 3D Structures. *Computer Graphics Forum (Eurographics STAR)* (2020).
- Ding-Jie Chen, Hwann-Tzong Chen, and Long-Wen Chang. 2012. Video object cosegmentation. In *Proceedings of the 20th ACM international conference on Multimedia*.
- Runnan Chen, Xinge Zhu, Nenglun Chen, Wei Li, Yuxin Ma, Ruigang Yang, and Wengping Wang. 2022. Zero-shot point cloud segmentation by transferring geometric primitives. *arXiv* (2022).
- Zhiqin Chen, Vladimir G. Kim, Matthew Fisher, Noam Aigerman, Hao Zhang, and Siddhartha Chaudhuri. 2021. DECOR-GAN: 3D Shape Detailization by Conditional Refinement. *CVPR* (2021).
- Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. 2020. BSP-Net: Generating Compact Meshes via Binary Space Partitioning. *CVPR* (2020).
- Zhiqin Chen, Kangxue Yin, Matthew Fisher, Siddhartha Chaudhuri, and Hao Zhang. 2019. BAE-NET: Branched Autoencoder for Shape Co-Segmentation. *ICCV* (2019).
- Zhiqin Chen and Hao Zhang. 2019. Learning Implicit Fields for Generative Shape Modeling. *CVPR* (2019).
- Dale Decatur, Itai Lang, and Rana Hanocka. 2023. 3D Highlighter: Localizing regions on 3D shapes via text descriptions. In *CVPR*.
- Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. 2023. Objaverse: A universe of annotated 3d objects. In *CVPR*.
- Boyan Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. 2020. CvxNet: Learnable convex decomposition. In *CVPR*.
- Yu Deng, Jiaolong Yang, and Xin Tong. 2021. Deformed implicit field: Modeling 3d shapes with learned dense correspondence. In *CVPR*. 10286–10296.
- Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. 2019. Learning elementary structures for 3D shape generation and matching. *NeurIPS* (2019).
- Runyu Ding, Jihang Yang, Chuhui Xue, Wenqing Zhang, Song Bai, and Xiaojuan Qi. 2023. PLA: Language-driven open-vocabulary 3D scene understanding. In *CVPR*.
- Zhiwen Fan, Peihao Wang, Yifan Jiang, Xinyu Gong, Dejia Xu, and Zhangyang Wang. 2022. NeRF-SOS: Any-view self-supervised object segmentation on complex scenes. In *ICLR*.
- Xiao Fu, Shangzhan Zhang, Tianrun Chen, Yichong Lu, Lanyun Zhu, Xiaowei Zhou, Andreas Geiger, and Yiyi Liao. 2022. Panoptic NeRF: 3D-to-2D label transfer for panoptic urban scene segmentation. In *3DV*.
- Rahul Goel, Dhawal Sirikonda, Saurabh Saini, and PJ Narayanan. 2023. Interactive segmentation of radiance fields. In *CVPR*.
- A. Golovinskiy and T. Funkhouser. 2009a. Consistent segmentation of 3D models. *Computers & Graphics (Proc. of SMI)* 33, 3 (2009), 262–269.
- Aleksey Golovinskiy and Thomas Funkhouser. 2009b. Consistent segmentation of 3D models. *Computers & Graphics* (2009).
- Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. 2011. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*.
- Yining Hong, Chunru Lin, Yilun Du, Zhenfang Chen, Joshua B Tenenbaum, and Chuang Gan. 2023. 3D concept learning and reasoning from multi-view images. In *CVPR*.
- Ruizhen Hu, Lubin Fan, and Ligang Liu. 2012. Co-segmentation of 3D shapes via subspace clustering. In *Comput. Graph. Forum*.
- Qixing Huang and Leonidas Guibas. 2013. Consistent shape maps via semidefinite programming. *Computer Graphics Forum (SGP)* 32, 5 (2013), 177–186.
- Qixing Huang, Vladlen Koltun, and Leonidas Guibas. 2011. Joint shape segmentation with linear programming. *ACM TOG* (2011).
- Xiaoyang Huang, Yi Zhang, Kai Chen, Teng Li, Wenjun Zhang, and Bingbing Ni. 2023. Learning Shape Primitives via Implicit Convexity Regularization. In *ICCV*. 3642–3651.
- Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language

- representation learning with noisy text supervision. In *ICML*.
- Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. 2019. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *CVPR*.
- Yuki Kawana, Yusuke Mukuta, and Tatsuya Harada. 2020. Neural star domain as primitive representation. *NeurIPS* (2020).
- Sihyeon Kim, Minseok Joo, Jaewon Lee, Juyeon Ko, Juhan Cha, and Hyunwoo J Kim. 2023. Semantic-Aware Implicit Template Learning via Part Deformation Consistency. In *ICCV*. 593–603.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: a method for stochastic optimization. In *ICLR*.
- Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. 2022. Decomposing NeRF for editing via feature field distillation. *NeurIPS* (2022).
- Juil Koo, Ian Huang, Panos Achlioptas, Leonidas J Guibas, and Minhyuk Sung. 2022. PartGlot: Learning shape part segmentation from language reference games. In *CVPR*.
- Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. 2022. Panoptic neural fields: A semantic object-aware neural scene representation. In *CVPR*.
- Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and Rene Ranftl. 2022a. Language-driven Semantic Segmentation. In *ICLR*.
- Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. 2022b. Grounded language-image pre-training. In *CVPR*.
- Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su. 2023. PartSLIP: Low-shot part segmentation for 3D point clouds via pretrained image-language models. In *CVPR*.
- Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. 2019. Neural volumes: learning dynamic renderable volumes from images. *ACM TOG* (2019).
- Min Meng, Jiazhui Xia, Jun Luo, and Ying He. 2013. Unsupervised co-segmentation for 3D shapes using iterative multi-label optimization. *CAD* (2013).
- Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. 2019. Occupancy Networks: Learning 3D Reconstruction in Function Space. In *CVPR*.
- Björn Michele, Alexandre Boulch, Gilles Puy, Maxime Bucher, and Renaud Marlet. 2021. Generative zero-shot learning for semantic segmentation of 3D point clouds. In *3DV*.
- Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*.
- Niloy Mitra, Michael Wand, Hao Zhang, Daniel Cohen-Or, and Martin Bokeloh. 2013. Structure-aware shape processing. In *SIGGRAPH Asia Course*.
- Chengjie Niu, Manyi Li, Kai Xu, and Hao Zhang. 2022. RIM-Net: Recursive implicit fields for unsupervised learning of hierarchical shape structures. In *CVPR*.
- Despoina Paschalidou, Luc Van Gool, and Andreas Geiger. 2020. Learning unsupervised hierarchical part decomposition of 3D objects from a single rgb image. In *CVPR*.
- Despoina Paschalidou, Angelos Katharopoulos, Andreas Geiger, and Sanja Fidler. 2021. Neural Parts: Learning expressive 3D shape abstractions with invertible neural networks. In *CVPR*.
- Despoina Paschalidou, Ali Osman Ulusoy, and Andreas Geiger. 2019. Superquadrics revisited: Learning 3D shape parsing beyond cuboids. In *CVPR*.
- Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. 2023. Openscene: 3d scene understanding with open vocabularies. In *CVPR*. 815–824.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*.
- Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. *NeurIPS* (2017).
- Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS* (2022).
- Ariel Shamir. 2008. A survey on mesh segmentation techniques. In *Comput. Graph. Forum*.
- Zhenyu Shu, Chengwu Qi, Shiqing Xin, Chao Hu, Li Wang, Yu Zhang, and Ligang Liu. 2016. Unsupervised 3D shape segmentation and co-segmentation via deep learning. *CAGD* (2016).
- Qingyao Shuai, Chi Zhang, Kaizhi Yang, and Xuejin Chen. 2023. DPF-Net: Combining Explicit Shape Priors in Deformable Primitive Field for Unsupervised Structural Reconstruction of 3D Objects. In *ICCV*. 14321–14329.
- Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulò, Norman Müller, Matthias Nießner, Angela Dai, and Peter Kortschieder. 2023. Panoptic lifting for 3D scene understanding with neural fields. In *CVPR*.
- Oana Sidi, Oliver van Kaick, Yanir Kleiman, Hao Zhang, and Daniel Cohen-Or. 2011. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. *ACM TOG* (2011).
- Chun-Yu Sun, Qian-Fang Zou, Xin Tong, and Yang Liu. 2019. Learning adaptive hierarchical cuboid abstractions of 3D shape collections. *ACM TOG* (2019).
- Konstantinos Tertikas, Despoina Paschalidou, Boxiao Pan, Jeong Joon Park, Mikaela Angelina Uy, Ioannis Emiris, Yannis Avrithis, and Leonidas Guibas. 2023. Generating Part-Aware Editable 3D Shapes Without 3D Supervision. In *CVPR*.
- Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. 2022. Neural Feature Fusion Fields: 3D distillation of self-supervised 2D image representations. In *3DV*.
- Shubham Tulsiani, Hao Su, Leonidas J Guibas, Alexei A Efros, and Jitendra Malik. 2017. Learning shape abstractions by assembling volumetric primitives. In *CVPR*.
- Oliver van Kaick, Kai Xu, Hao Zhang, Yanzhen Wang, Shuyang Sun, Ariel Shamir, and Daniel Cohen-Or. 2013. Co-Hierarchical Analysis of Shape Structures. *ACM TOG* 32, 4 (2013), Article 69.
- Sara Vicente, Carsten Rother, and Vladimir Kolmogorov. 2011. Object cosegmentation. In *CVPR*. 2217–2224.
- Suhani Vora, Nohu Radwan, Klaus Greff, Henning Meyer, Kyle Genova, Mehdi SM Sajjadi, Etienne Pot, Andrea Tagliasacchi, and Daniel Duckworth. 2021. NeSF: Neural semantic fields for generalizable semantic segmentation of 3D scenes. *arXiv* (2021).
- Kai Xu, Vladimir G. Kim, Qixing Huang, Niloy Mitra, and Evangelos Kalogerakis. 2016. Data-Driven Shape Analysis and Processing. In *SIGGRAPH Asia Course*.
- Kai Xu, Honghua Li, Hao Zhang, Daniel Cohen-Or, Yueshan Xiong, and Zhi-Quan Cheng. 2010. Style-content separation by anisotropic part scales. *ACM TOG* (2010).
- Cheng-Kun Yang, Yung-Yu Chuang, and Yen-Yu Lin. 2021. Unsupervised point cloud object co-segmentation by co-contrastive learning and mutual attention sampling. In *ICCV*.
- Cheng-Kun Yang, Ji-Jia Wu, Kai-Syun Chen, Yung-Yu Chuang, and Yen-Yu Lin. 2022. An MIL-derived transformer for weakly supervised point cloud segmentation. In *CVPR*.
- Kaizhi Yang and Xuejin Chen. 2021. Unsupervised learning for cuboid shape abstraction via joint segmentation from point clouds. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–11.
- Li Yi, Vladimir G Kim, Duygu Ceylan, I Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, Leonidas Guibas, et al. 2016. A scalable active framework for region annotation in 3D shape collections. *ACM TOG* (2016).
- Haotian Zhang, Pengchuan Zhang, Xiaowei Hu, Yen-Chun Chen, Liunian Li, Xiyang Dai, Lijuan Wang, Lu Yuan, Jenq-Neng Hwang, and Jianfeng Gao. 2022. GLIPv2: Unifying localization and vision-language understanding. *NeurIPS* (2022).
- Zerong Zheng, Tao Yu, Qionghai Dai, and Yebin Liu. 2021. Deep implicit templates for 3d shape representation. In *CVPR*. 1429–1439.
- Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. 2021. In-place scene labelling and understanding with implicit scene representation. In *ICCV*.
- Chenyang Zhu, Kai Xu, Siddhartha Chaudhuri, Li Yi, Leonidas J Guibas, and Hao Zhang. 2020. AdaCoSeg: Adaptive shape co-segmentation with group consistency loss. In *CVPR*.

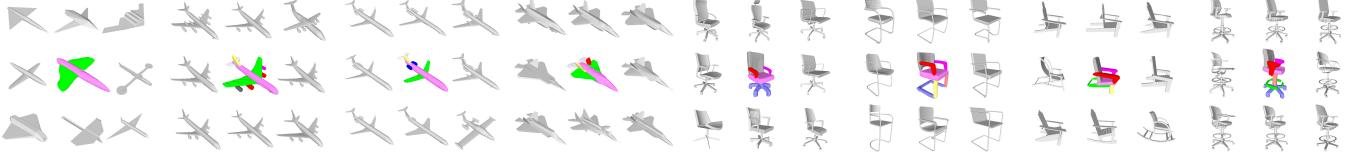


Figure 8: Shape clustering using the part existence score. Each 3x3 sub-figure shows a clustered group. The center of each sub-figure shows a reconstructed shape representing the parts that shapes in this group should have; the rest shows the first few shapes in the group.

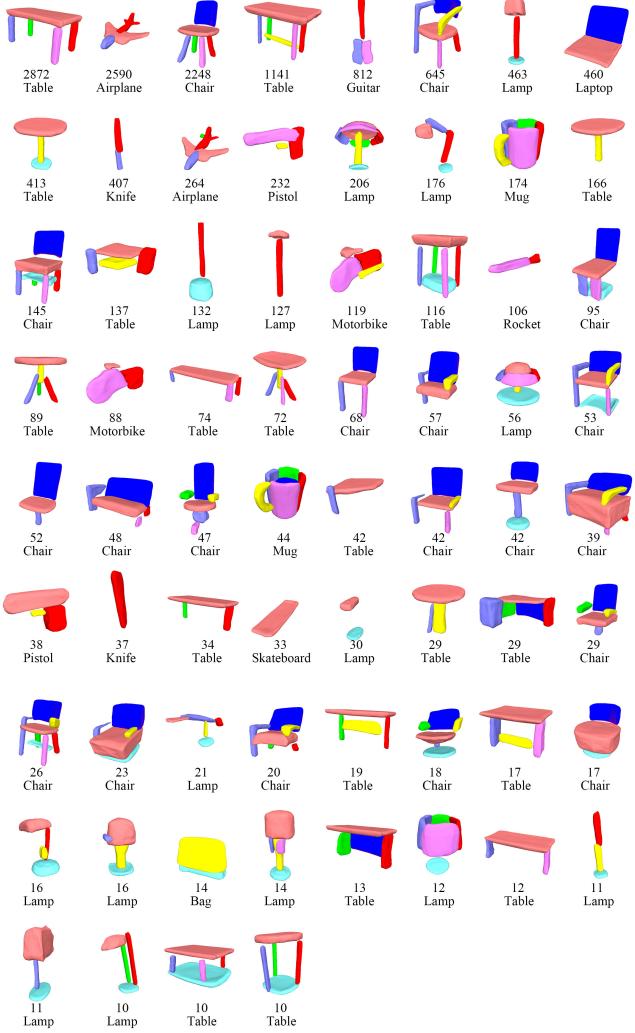


Figure 9: Shape clusters when trained on all the categories in ShapeNet Part dataset. We show all groups that contain at least 10 shapes. For each group, we show a representative reconstructed shape, the number of shapes it contains, and its semantic label for computing the numbers in Table 3.

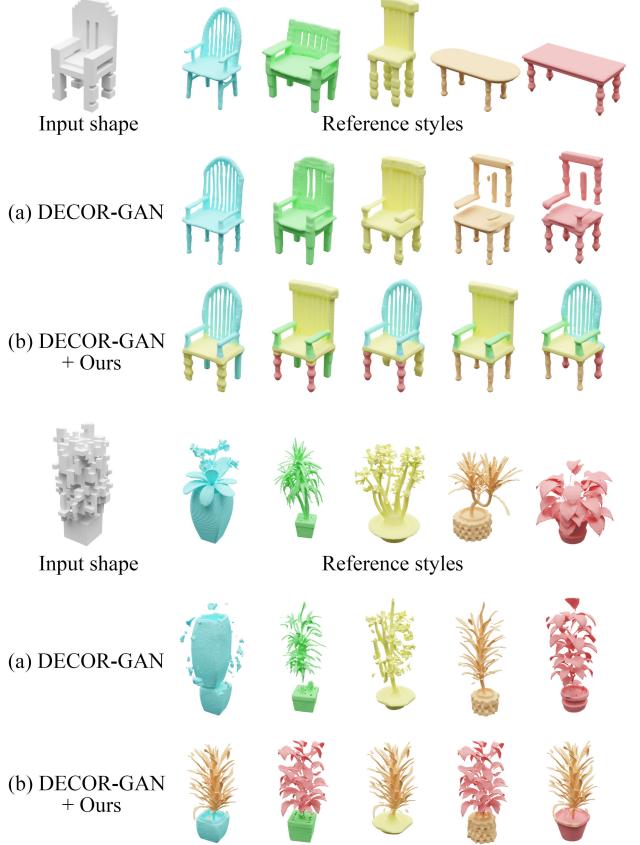


Figure 10: Part-level shape detailization on chair-table and plant. We use different colors to distinguish different geometric styles. To generate geometric details on an input coarse voxel shape, DECOR-GAN [Chen et al. 2021] can only adopt the geometric style of one of the reference shapes, while our method enables it to apply different styles to different parts.

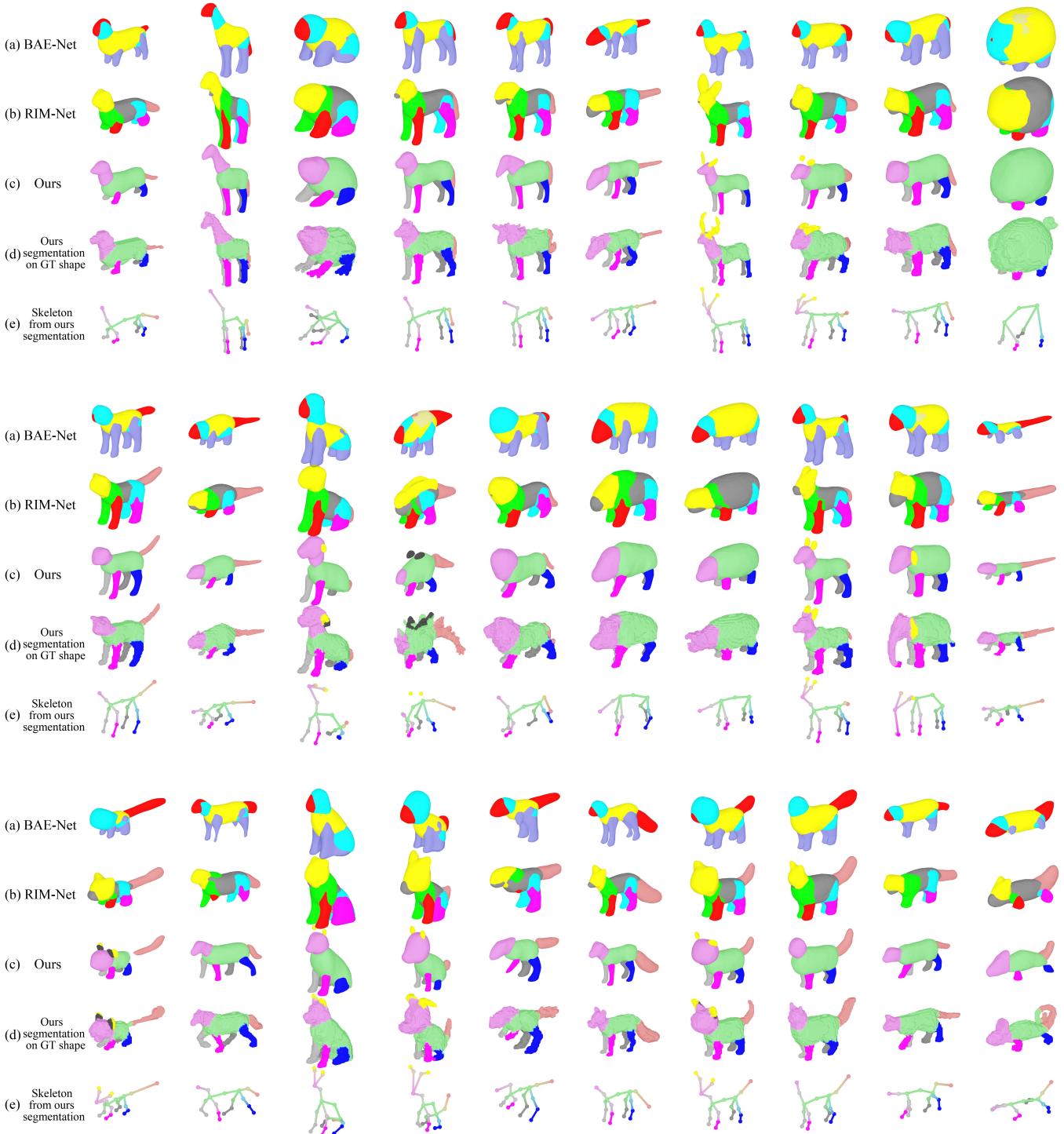


Figure 11: Qualitative results on shape segmentation on the quadruped animal subset of Objaverse, compared with other methods. Within the same category, same color indicates the parts are from the same branch of the network, thus are considered to be corresponded. Notice how semantically consistent our segmented parts are, when compared with other methods. We also show the skeletons built upon our segmentation.

DAE-Net: Deforming Auto-encoder for unsupervised shape co-segmentation
 (Supplementary Material)

A SEGMENTATION TO SKELETON

To build a skeleton, we first use our segmentation method to segment a 64^3 voxel representation of the shape. We then identify all the parts in this shape. Note that for the same semantic part, we treat each connected component as an individual part, e.g., horns in Figure 12. We then create a node at the center of each part, denoted as “part nodes”. We also create a node whenever two parts meet, and consider these nodes “joint nodes”. The joint nodes are connected to the corresponding part nodes. We also create additional part nodes according to some heuristics: we create one node to join the front legs, one to join the back legs, and one node at the far end of each limb, as shown in Figure 12 (b). After creating the initial skeleton, we treat the positions of all part nodes as optimizable parameters, and minimize the symmetric chamfer distance between the points sampled on the skeleton and the points sampled inside the ground truth voxel shape. After optimization with gradient decent, we obtain the final skeleton, as shown in Figure 12 (c).

B MORE QUALITATIVE RESULTS

We show qualitative results on DFAUST in Figure 13 and ShapeNet Part dataset in Figure 14 15 16 17 18.

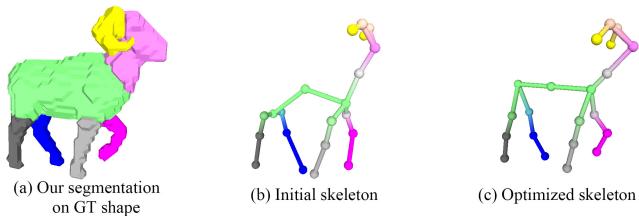


Figure 12: Building a skeleton from a segmented shape.

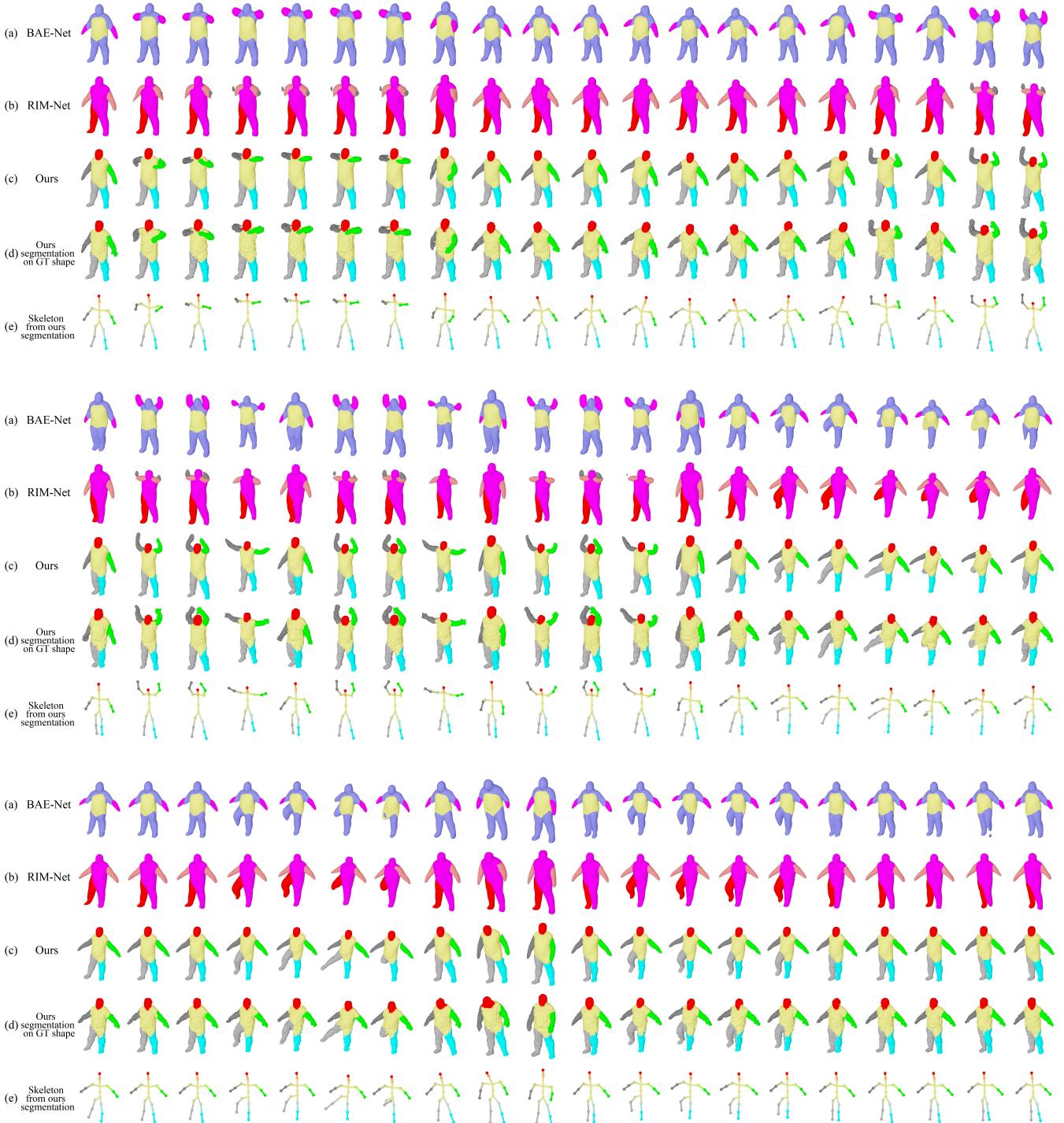


Figure 13: Qualitative results on shape segmentation on DFAUST, compared with other methods. Within the same category, same color indicates the parts are from the same branch of the network, thus are considered to be corresponded. We also show the skeletons built upon our segmentation.

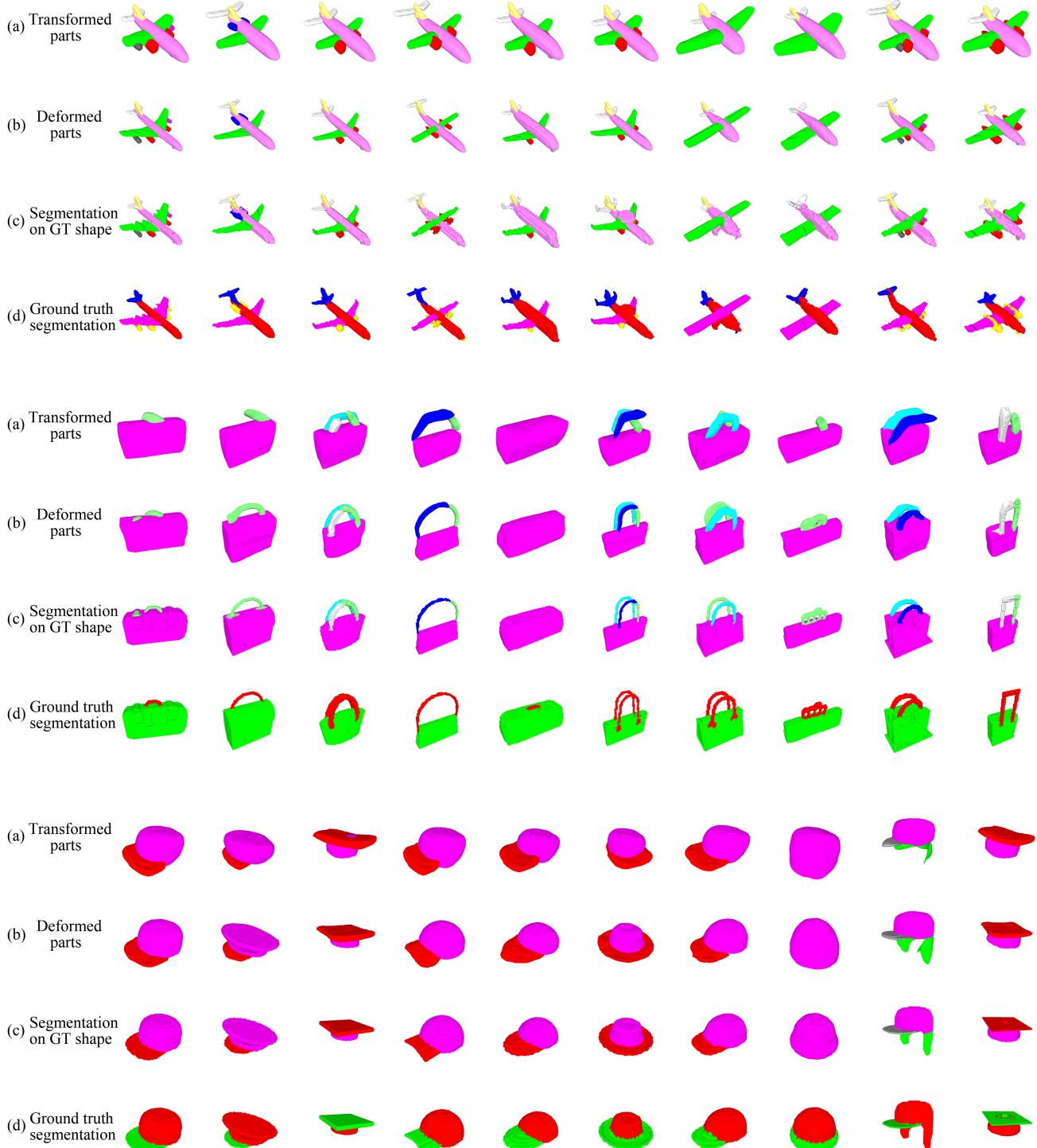


Figure 14: Qualitative results on shape segmentation on the airplane, bag, and cap categories of the ShapeNet Part dataset. Within the same category, same color indicates the parts are from the same branch of the network, thus are considered to be corresponded.

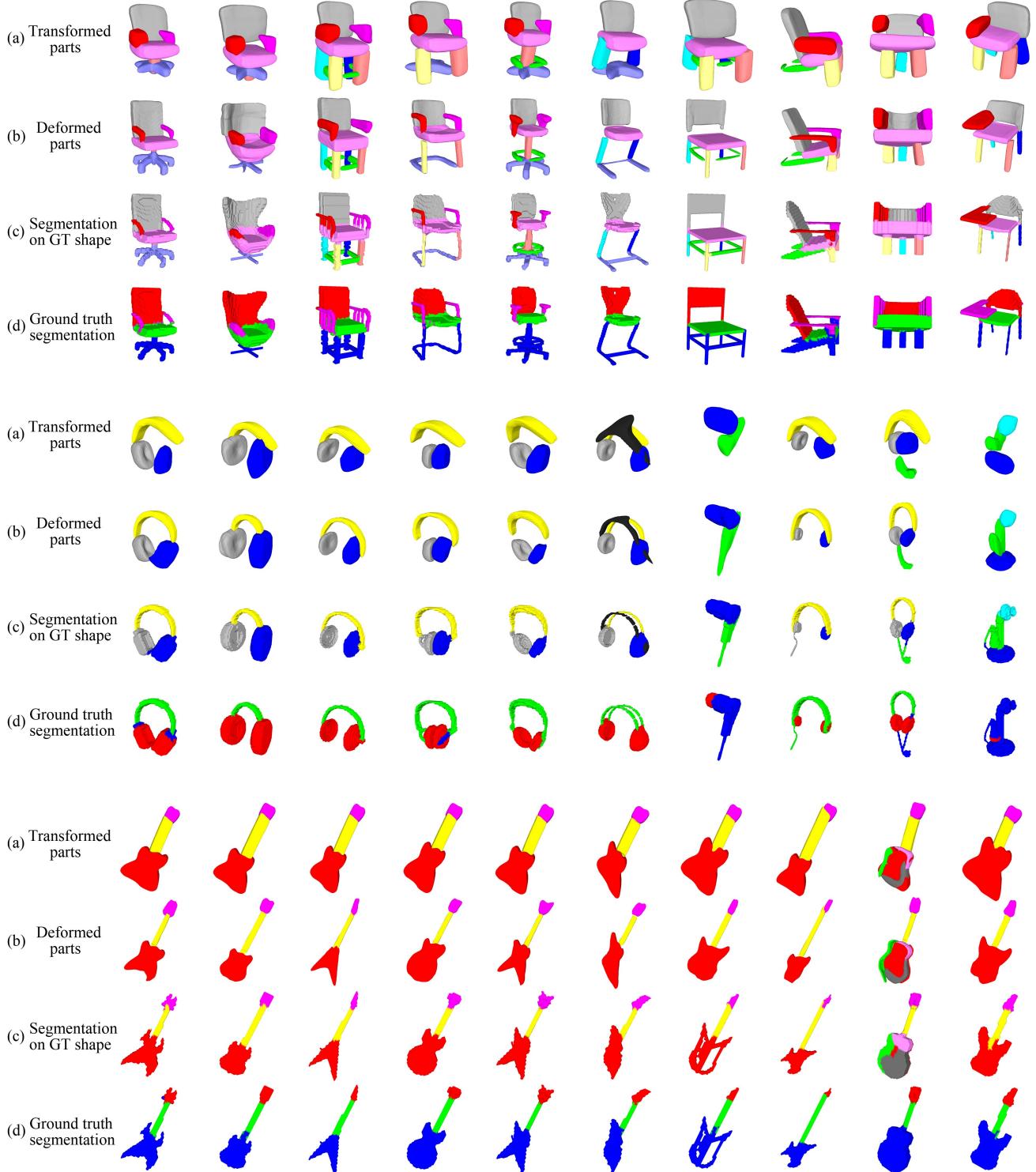


Figure 15: Qualitative results on shape segmentation on the chair, earphone, and guitar categories of the ShapeNet Part dataset. Within the same category, same color indicates the parts are from the same branch of the network, thus are considered to be corresponded.

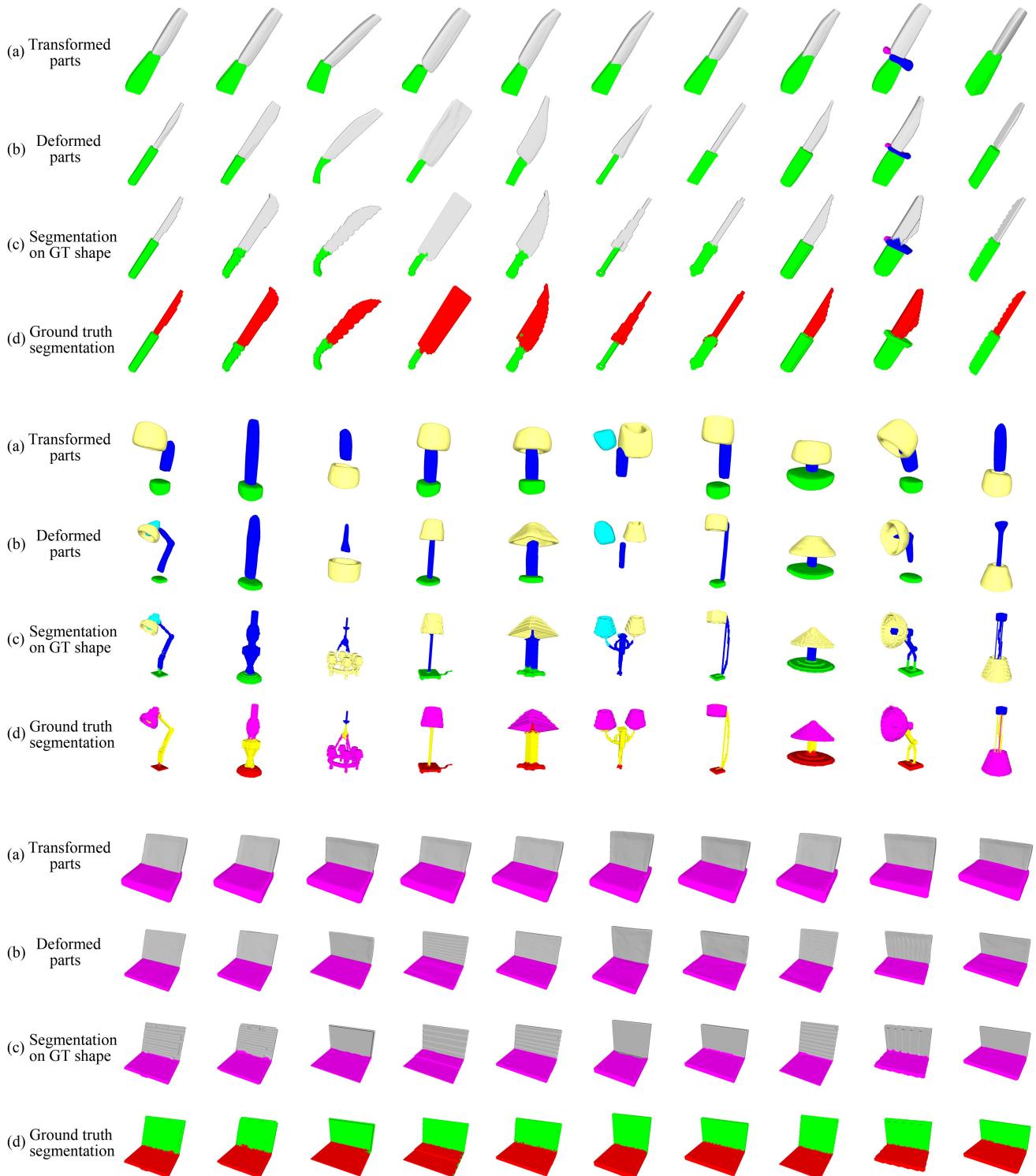


Figure 16: Qualitative results on shape segmentation on the knife, lamp, and laptop categories of the ShapeNet Part dataset. Within the same category, same color indicates the parts are from the same branch of the network, thus are considered to be corresponded.

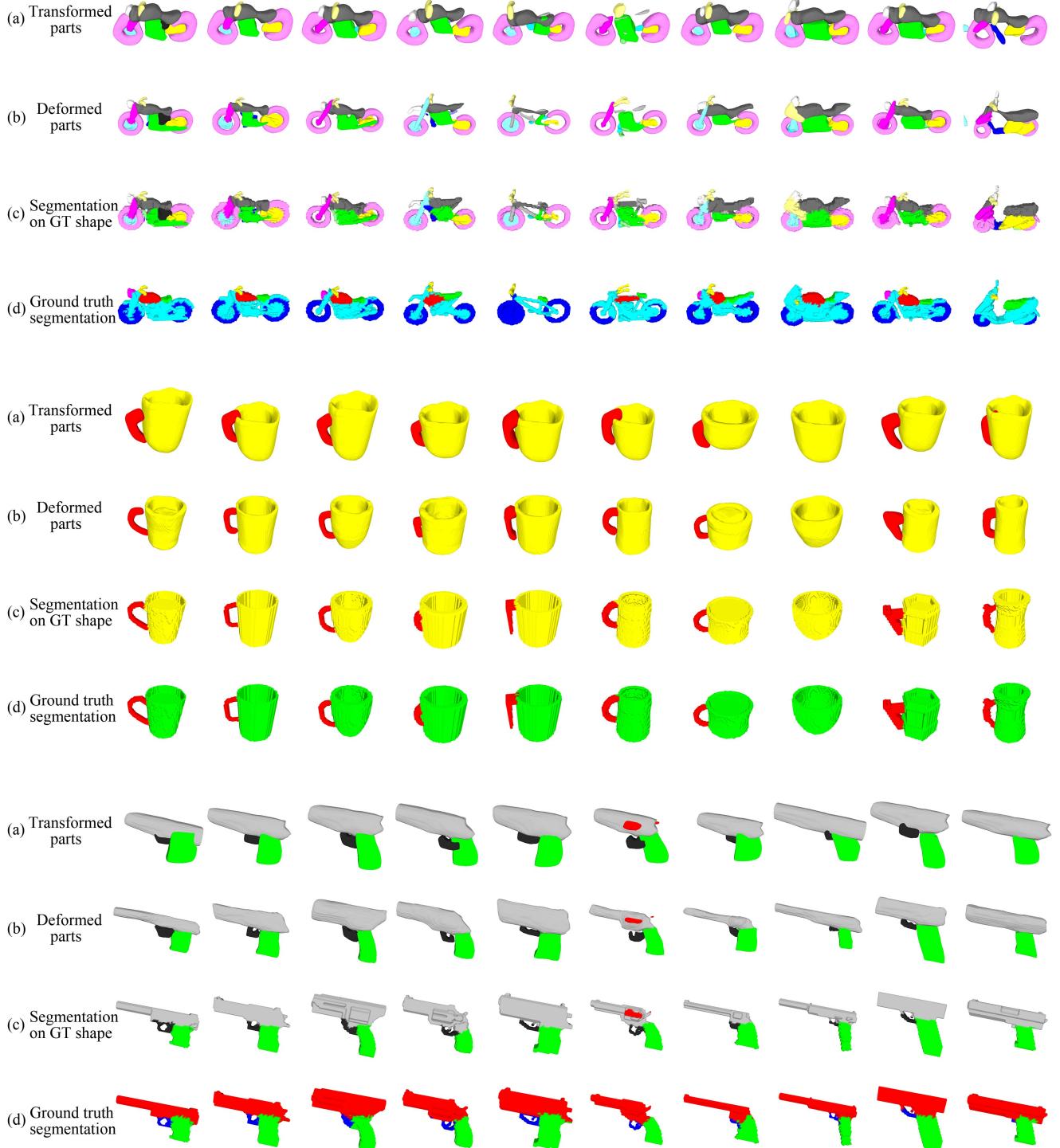


Figure 17: Qualitative results on shape segmentation on the motorbike, mug, and pistol categories of the ShapeNet Part dataset. Within the same category, same color indicates the parts are from the same branch of the network, thus are considered to be corresponded.

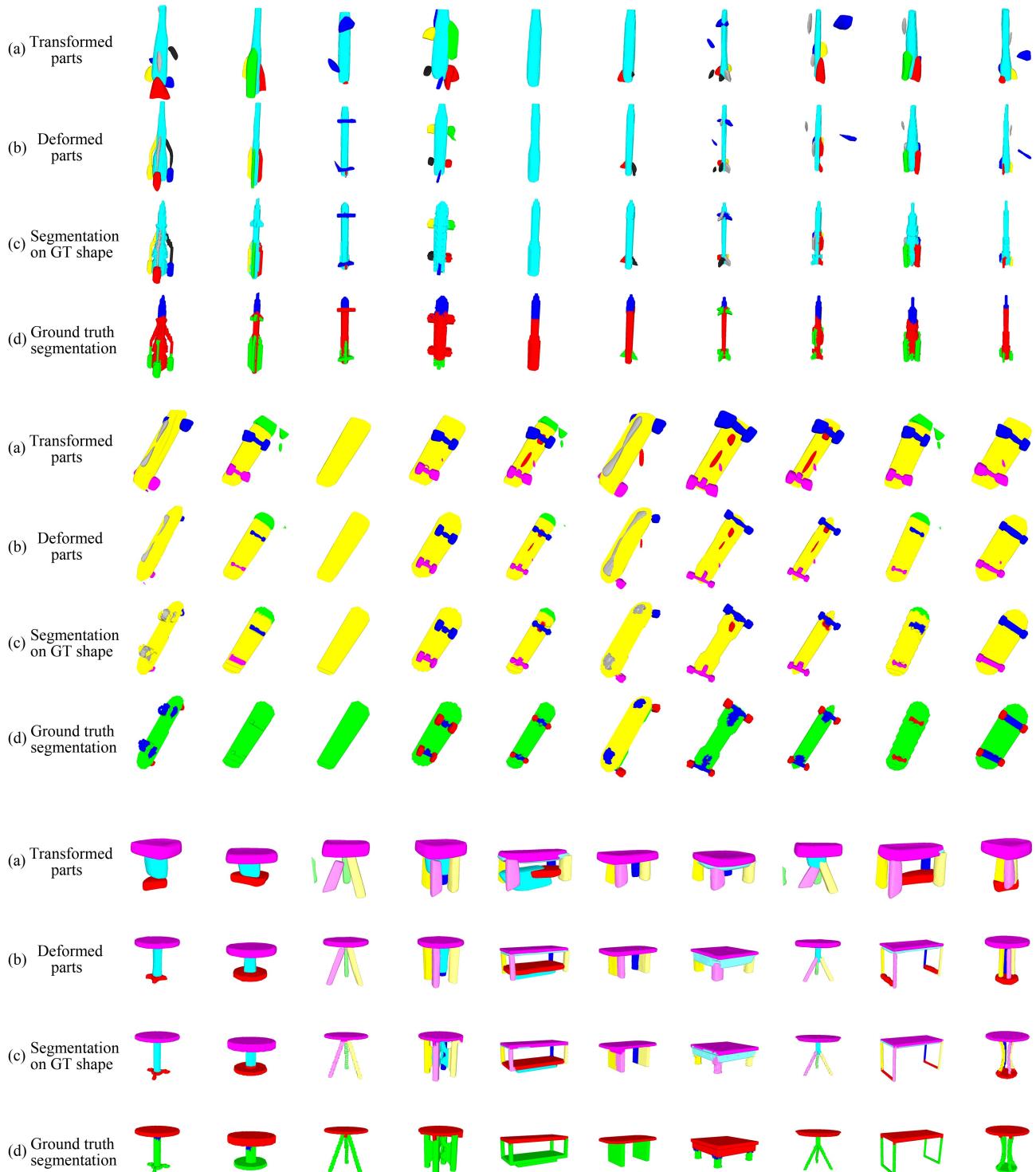


Figure 18: Qualitative results on shape segmentation on the rocket, skateboard, and table categories of the ShapeNet Part dataset. Within the same category, same color indicates the parts are from the same branch of the network, thus are considered to be corresponded.