Composite Shape Modeling via Latent Space Factorization
Supplementary Material

Anastasia Dubrovina[1] Fei Xia[1] Panos Achlioptas[1] Mira Shalah[1]
Raphaël Groscot[2] Leonidas Guibas[1]
[1]Stanford University [2]PSL Research University

1. Decomposer-Composer architecture

The Decomposer consists of a whole-shape encoder and K projection layers, where K is the number of semantic part labels. The architecture of the whole-shape encoder is given in Table 1. The projection layers are implemented as fully connected layers, with 100 outputs, where 100 is the dimension of the embedding space.

The Composer consists of a shared part decoder, and a Spatial Transformer Network (STN). The architecture of the part decoder is given in Table 2. STN, similar to the original design in [1], consists of a localization sub-network, and a re-sampling module. The re-sampling module uses trilinear interpolation, and does not have learned parameters. The localization network receives both K stacked decoded parts, and the sum of part embeddings, of dimension 100. First, the two inputs are separately processed: the stacked decoded parts - using two FC layers with 256 outputs; the sum of part encodings - using one FC layer with 128 outputs. The two results are then concatenated into a single 384-dimensional vector, and processed with two additional FC layers with 128 and 12 K outputs (K times 12 affine transformation parameters), respectively. All FC layers, except for the last one, are followed by ReLU layers, and dropout layers with keep probability of 0.7.

Type	Kernel	Stride	Outputs	Output size
conv.	$5 \times 5 \times 5$	$1 \times 1 \times 1$	16	32^3
conv.	$5 \times 5 \times 5$	$2 \times 2 \times 2$	32	16^3
conv.	$5 \times 5 \times 5$	$2 \times 2 \times 2$	64	8^3
conv.	$3 \times 3 \times 3$	$2 \times 2 \times 2$	128	4^3
conv.	$3 \times 3 \times 3$	$2 \times 2 \times 2$	256	2^3
FC	-	-	100	1

Table 1: Whole-shape encoder (Decomposer) architecture. Each convolution layer (”conv.”) is followed by a Rectified Linear Unit (ReLU) layer, and a batch normalization layer. The last is a fully-connected layer (”FC”).

2. Binary shape classifier architecture

In the evaluation of the proposed method, we used a binary classifier to estimate the quality of assembly and how realistic the resulting shapes were (see Section 4.4.3 in the paper for details). The architecture of the classifier is shown in Table 3.

3. Latent space and projection matrix analysis

Latent space Figure 1 visualizes the structure of our learned part latent space, for chair shapes from the ShapeNet, using the T-SNE algorithm [2], and illustrates the clear separation into different semantic part subspaces.

Projection matrix analysis Figure 2 shows the projection matrices, learned for the chair class, sum of projection matrices, and the plot of their singular values. The proposed method succeeds to obtain a set of projection matrices which approximately sum to an identity, and have a partition of the identity loss (Eq. (3) in the paper) of the order of one, for a hundred-dimensional latent space and four semantic subspaces. While $\{P_i\}_{i=1}^4$ are full-rank and not strictly orthogonal projection matrices, the plot of their singular values shows that their effective ranks are significantly lower than the latent space dimension. This is also

Type	Kernel	Stride	Outputs	Output size
FC	-	-	256	2^3
deconv.	$3 \times 3 \times 3$	$2 \times 2 \times 2$	128	4^3
deconv.	$3 \times 3 \times 3$	$2 \times 2 \times 2$	64	8^3
deconv.	$5 \times 5 \times 5$	$2 \times 2 \times 2$	32	16^3
deconv.	$5 \times 5 \times 5$	$2 \times 2 \times 2$	16	16^3
conv.	$5 \times 5 \times 5$	$1 \times 1 \times 1$	1	32^3

Table 2: Part decoder (Composer) architecture. The fully-connected layer (”FC”), and every de-convolution layer (”deconv.”), are followed by a Rectified Linear Unit (ReLU) layer, a batch normalization layer, and a dropout with keep probability 0.8.

Type	Kernel	Stride	Outputs	Output size
conv.	$6 \times 6 \times 6$	$2 \times 2 \times 2$	32	16^3
conv.	$6 \times 6 \times 6$	$2 \times 2 \times 2$	32	8^3
conv.	$4 \times 4 \times 4$	$2 \times 2 \times 2$	64	4^3
conv.	$2 \times 2 \times 2$	$2 \times 2 \times 2$	64	2^3
conv.	$2 \times 2 \times 2$	$2 \times 2 \times 2$	128	1
DO (0.5)	-	-	128	1
FC_1	-	-	128	1
FC_2	-	-	64	1
FC_3	-	-	2	1

Table 3: Architecture of the binary classifier. Each convolution layer (”conv.”) is followed by a Rectified Linear Unit (ReLU) and a batch normalization layers. Dropout layer (”DO”) has a keep probability of 0.5. The fully-connected layers FC_1 and FC_2 are followed by batch normalization and ReLU layers. The classifier produces binary output.

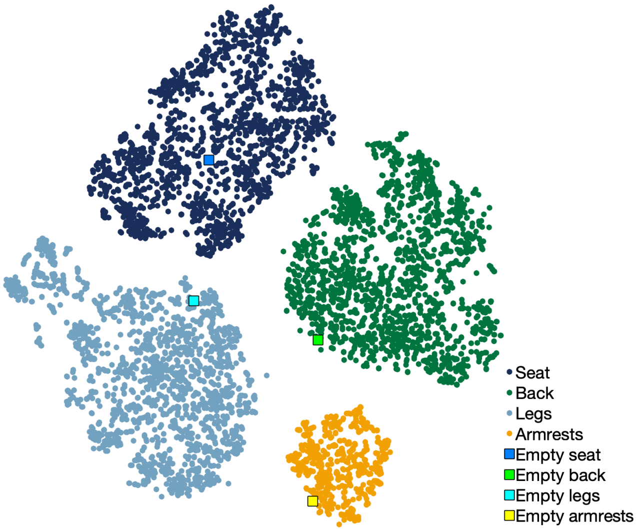
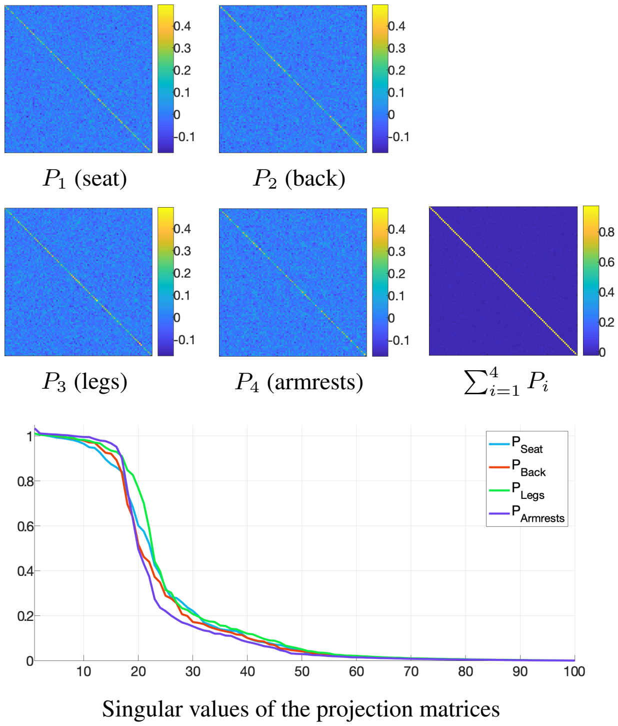


Figure 1: T-SNE [2] visualization of the produced embedding space, using both train and test shape embedding coordinates. The ”empty” part coordinates correspond to the embedding coordinates of non-existing semantic parts.

in line with the excellent separation into non-overlapping subspaces produced by these projection matrices.

4. Shape-from-random-parts synthesis

Figure 3 presents the result of assembling shapes from random parts, for chair, table, guitar and airplane shape classes. For this experiment, we worked with shapes from the test set, using batches of the size of the number of semantic parts in the shapes: four shapes in a batch for chairs and airplanes, three - for guitars, and two - for tables. We synthesized corresponding new shapes by, first, creating new part encoding sets, by randomly mixing part encodings of the input shapes, ensuring that no two encodings in the new set come from the same input shape; We then reconstructed the shapes using the Composer. The results in Figure 3 illustrate the ability of the proposed method to combine parts from different shapes, and scale and place them so that the resulting shape looks realistic. The method was applied on *unlabeled* input shapes. The results also demonstrate limitations of the proposed approach: occasionally, parts are not faithfully reconstructed (*e.g.*, legs of the rightmost chair in the second row do not resemble the legs of the source chair - second from the right in the first row), or the produced shape is disconnected (legs of the rightmost chair in the second row are not connected the seat).


P_1 (seat) P_2 (back)

P_3 (legs) P_4 (armrests) $\sum_{i=1}^{4} P_i$

Singular values of the projection matrices

Figure 2: Projection matrix analysis. Two upper rows present the obtained projection matrices, and their sum. The bottom row shows the singular values of the matrices.

5. Full and partial shape interpolation in the embedding space

Figure 4 presents additional examples of shapes obtained by linear interpolation of the input shapes’ embedding coordinates, and reconstructed from these interpolated embeddings using the Composer. The figure presents the ground truth shapes, their decoded versions, and eight interpolated shapes. Note that the proposed network operates on *unlabeled* input shapes, and produces gradual and plausible interpolations of pairs of shapes.

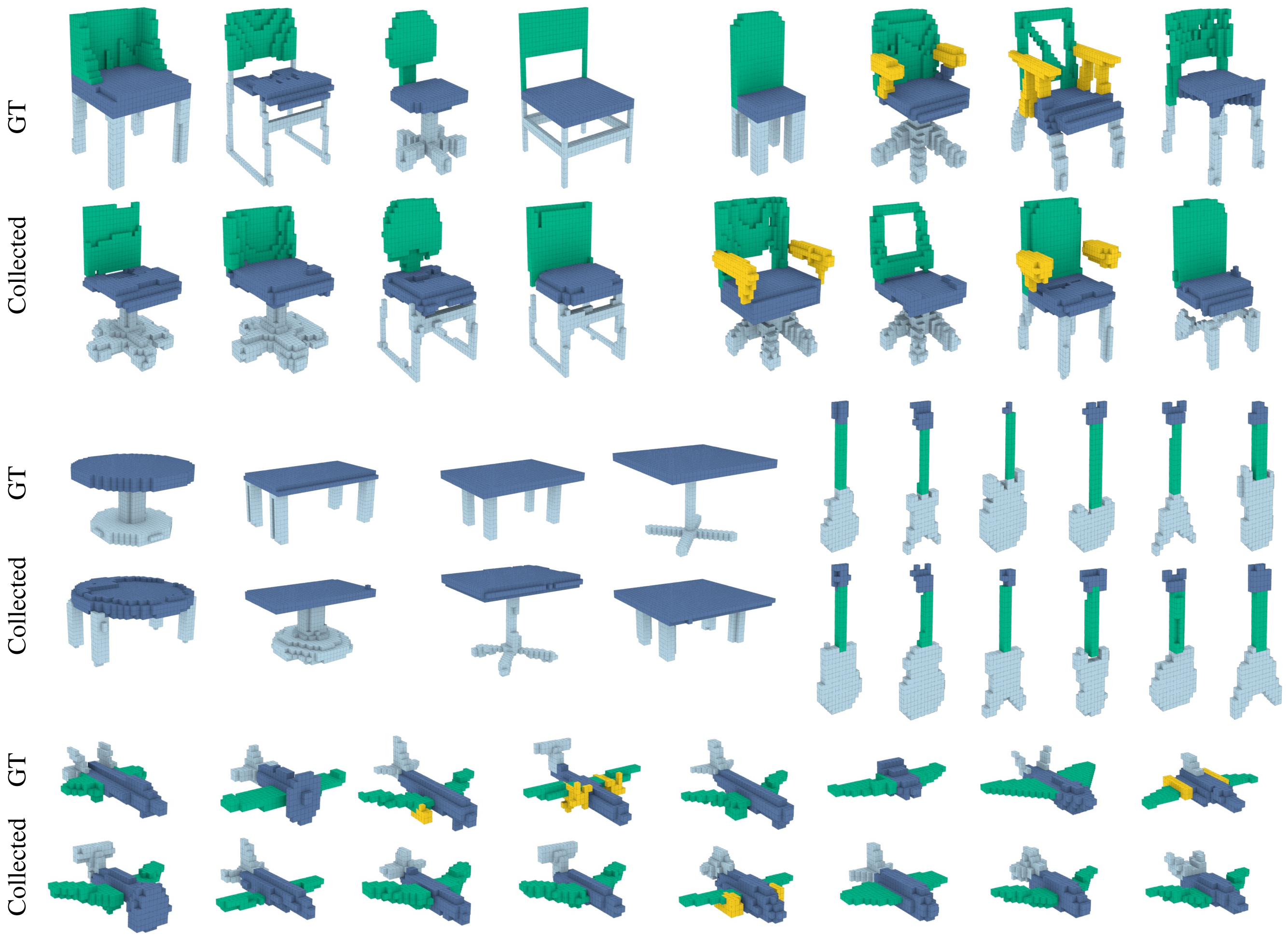


Figure 3: Synthesis-from-parts example. For every batch of 4 chairs, the top row shows the ground truth (GT) shapes, and the bottom row - shapes assembled by randomly picking parts from the GT shapes, such that no two parts come from the same GT shape, and assembled using the proposed approach. Unlabeled shapes were used as an input, and labeled GT shapes are shown for illustration purpose only.

Figure 5 presents examples of chair shapes obtained by linear interpolation of a single part, a functionality *unique* to the proposed approach. Specifically, given two shapes, we exchanged a single part, *e.g.*, a seat, between them, by changing the corresponding part embedding coordinates produced by the Decomposer. We then interpolated just these two embedding coordinates, and reconstructed new shapes from the interpolation result, together with the rest of the original part embedding coordinates, using the Composer. As illustrated by the results in Figure 5, the specified part changes gradually, from the source to the target part. The rest of the decoded parts remain visibly similar to the original ones, while still adapting to the change in the interpolated part - for example, the seat and the legs of the left chair in Figure 5, second row, become smaller as the back interpolation proceeds. Here again, the proposed network operates on *unlabeled* input shapes.

6. Ablation study visualization

Figures 6 and 7 present visual comparison between the results of the proposed method and the methods it was compare to in the ablation study. Figure 6 present the results of shape reconstruction, and Figure 7 - the result of shape assembly from random parts. We observe that the proposed method achieves most complete and realistically looking reconstruction results. Using fixed projection produces inferior part reconstruction results (by ”fixed projection” we mean dividing the embedding vector of the whole shape into pre-defined non-overlapping segments corresponding to different parts). So does the version without the STN in the Composer; There, the network fails to reconstruct

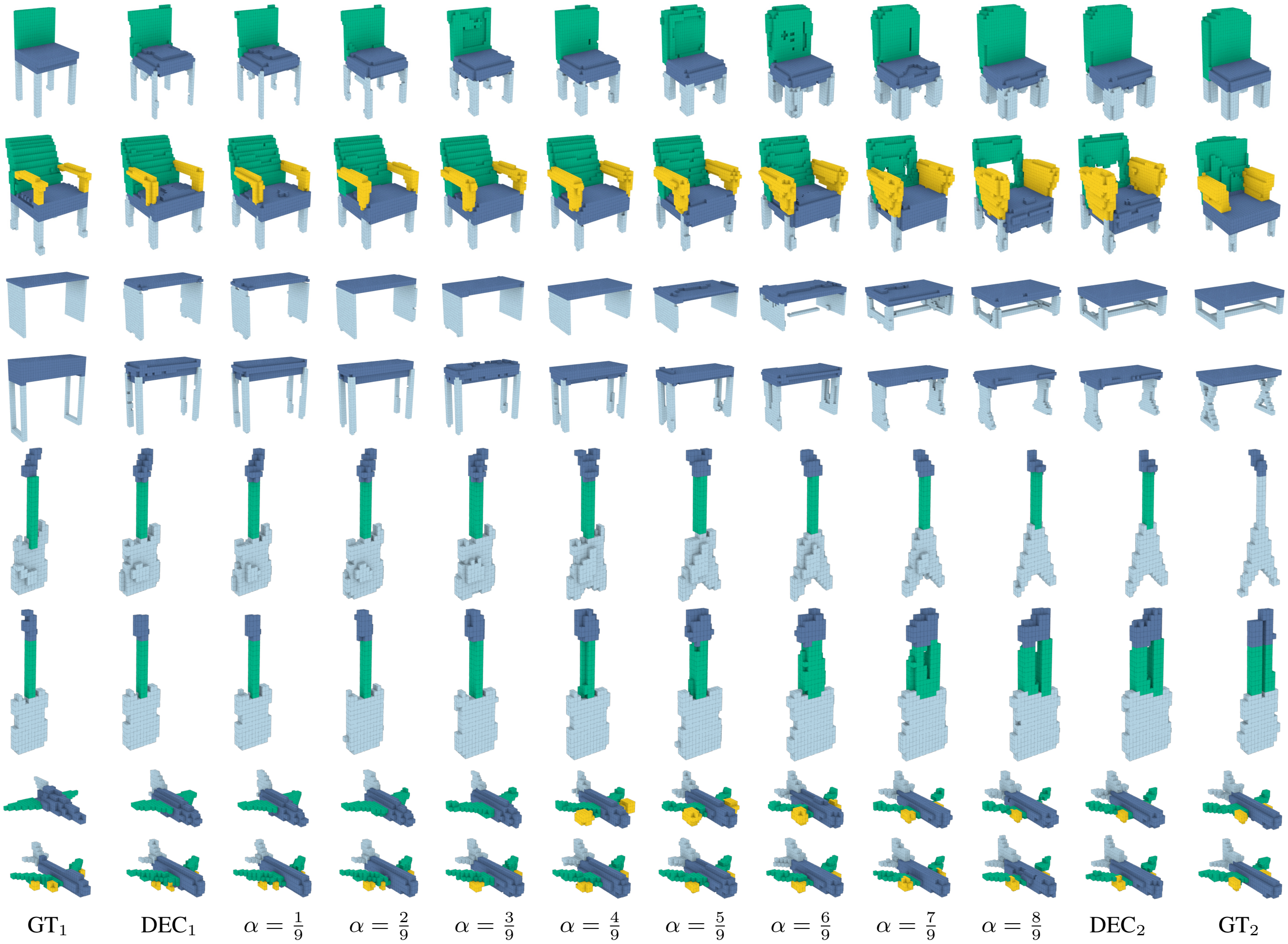


Figure 4: Example of a whole shape interpolation. Left and right are test models with ground truth segmentation. The rest of the results were obtained by linearly interpolating their embedding vectors (with the weight α), and reconstructing the shapes using the Composer network. Note that unlabeled shapes were used as an input.

small and fine shape parts, resulting in disconnected output shapes. Removing the cycle loss produces results with inferior part placement and part reconstruction quality.

Figure 7 also presents the results obtained with two competing methods, where the shape decomposition into parts and shape composition is performed using separate segmentation and placement networks (see Section 4.4.2 in the paper). We observe that neither placement with ComplementMe [3], nor with a Spatial Transformer Network, are able to produce plausible results when assembling shapes from random parts. We thus conclude that end-to-end training for shape decomposition and composition, performed by the proposed Decomposer and Composer, respectively, is essential for high quality reconstruction results. Note that, due to different experimental settings, not all method variations and competing methods use the same parts for shape assembly, which does not affect the conclusions above.

7. 64^3 reconstruction results

We re-trained the proposed Decomposer-Composer network with chair shapes from the ShapeNet, voxelized at $64 \times 64 \times 64$ resolution. The results in Figure 8 show that the network produces higher quality shape assembly results, at the expense of longer training time (4 days).

8. Additional comparisons

Figure 8 presents a comparison of the proposed method with Global-to-Local [4] and 3D-GAN [5] methods, on the shape reconstruction task. The proposed method significantly outperforms the 3D-GAN, and perform on-par with Global-to-Local method, while also offering the ability to perform per-part shape modelling, illustrated in Section 4 and 5, which 3D-GAN and Global-to-Local lack. SAGnet [6] doesn't have a public implementation we could compare

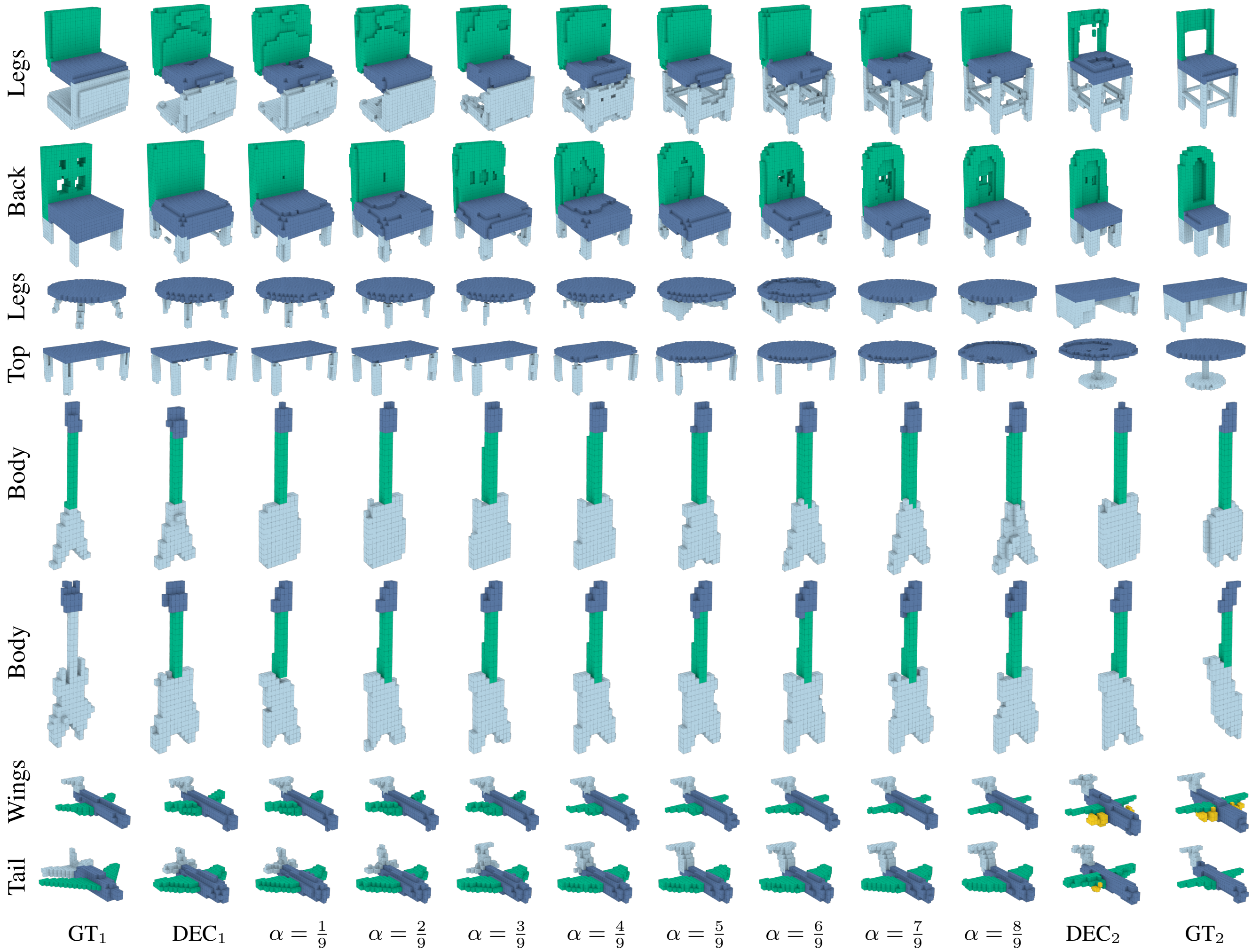


Figure 5: Example of a per-part shape interpolation. Left and right are test models with ground truth segmentation. The rest of the results were obtained by linearly interpolating a single part (stated on the left) in the left shape. Note that unlabeled shapes were used as an input. See the accompanying text in Section 5 for a detailed explanation.

to, but we expect it to perform on-par or somewhat better than the proposed method for shape synthesis, since it was trained using pre-segmented models. However, it too lacks the flexibility of part-based shape modeling.

9. Affine transformation analysis

Figure 9 present the comparison between the ground truth transformation parameters, and the parameters produced by our spatial transformer network for chair shapes, in the shape reconstruction and shape-from-part-assembly experiments. The notations used in Figure 9 assume that the transformation is given in homogeneous coordinates as

$$T = \begin{pmatrix} a_{11} & a_{12} & a_{13} & t_1 \\ a_{21} & a_{22} & a_{23} & t_2 \\ a_{31} & a_{32} & a_{33} & t_3 \\ 0 & 0 & 0 & 1, \end{pmatrix}, \tag{1}$$

where $\{a_{ij}\}_{i,j=1}^{3}$, are the affine transformation parameters, and $\{t_i\}_{i=1}^{3}$, are the translation parameters. The ground truth transformations in our dataset only down-scale and translate the centered and scaled parts of the shape. Therefore, in these transformations, $a_{11} = a_{22} = a_{33}$, and $a_{ij} = 0$ otherwise. We observe that the transformations produced by the spatial transformer network resemble the ground truth ones, and the cycle consistency requirement does not help the proposed method to learn more complex

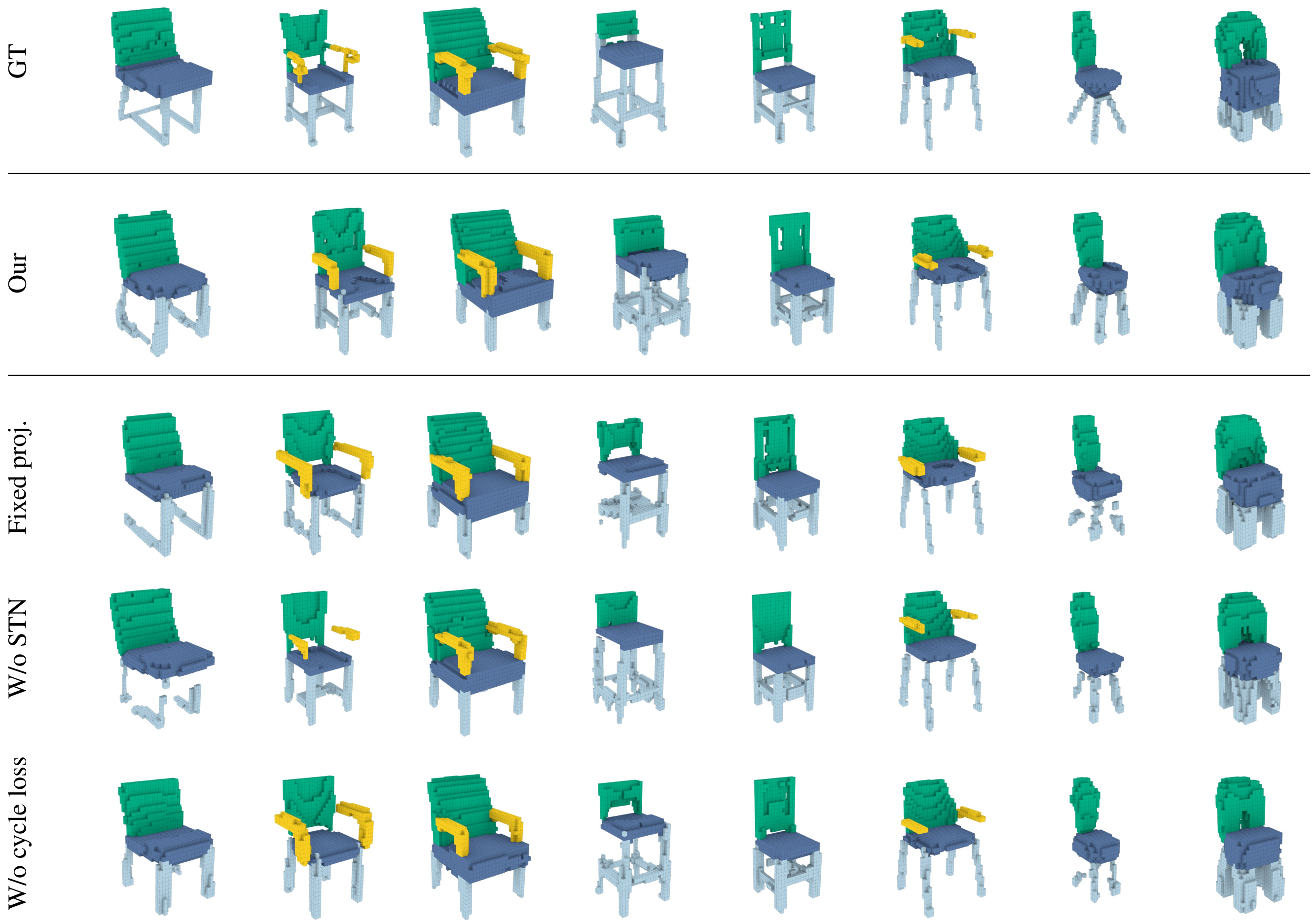


Figure 6: Ablation study: reconstruction result visualization. The top row shows input shapes with ground truth part labels. The following rows present the reconstruction result of the proposed method (Our), obtained with fixed projections (Fixed proj.), decoder without STN (W/o STN), and without cycle loss (W/o cycle loss).

affine transformations. Thus, while the proposed method is able to generate plausible shapes by exchanging parts or collecting parts at random, it does not yet fully exploit the capacity of full affine transformations, and may be expected to fail for shapes with more complex part arrangements than in the four classes of shapes used in our experiments. This also implies that, instead of affine transformations (12 parameters), our network could be trained to produce only non-uniform scaling and translation transformations (6 parameters). Furthermore, the transformations produced by the network have smaller variance that the original ones, but this still results in plausible shape reconstructions. We plan to investigate this further, and devise a method for fully utilizing affine and other types of transformations in future research.

References

[1] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015. 1

[2] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 1, 2

[3] Minhyuk Sung, Hao Su, Vladimir G Kim, Siddhartha Chaudhuri, and Leonidas Guibas. Complementme: weakly-supervised component suggestions for 3d modeling. *ACM Transactions on Graphics (TOG)*, 36(6):226, 2017. 4

[4] Hao Wang, Nadav Schor, Ruizhen Hu, Haibin Huang, Daniel Cohen-Or, and Hui Huang. Global-to-local generative model for 3d shapes. *ACM Transactions on Graphics (Proc. SIGGRAPH ASIA)*, 37(6):214:1214:10, 2018. 4, 8

[5] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling.

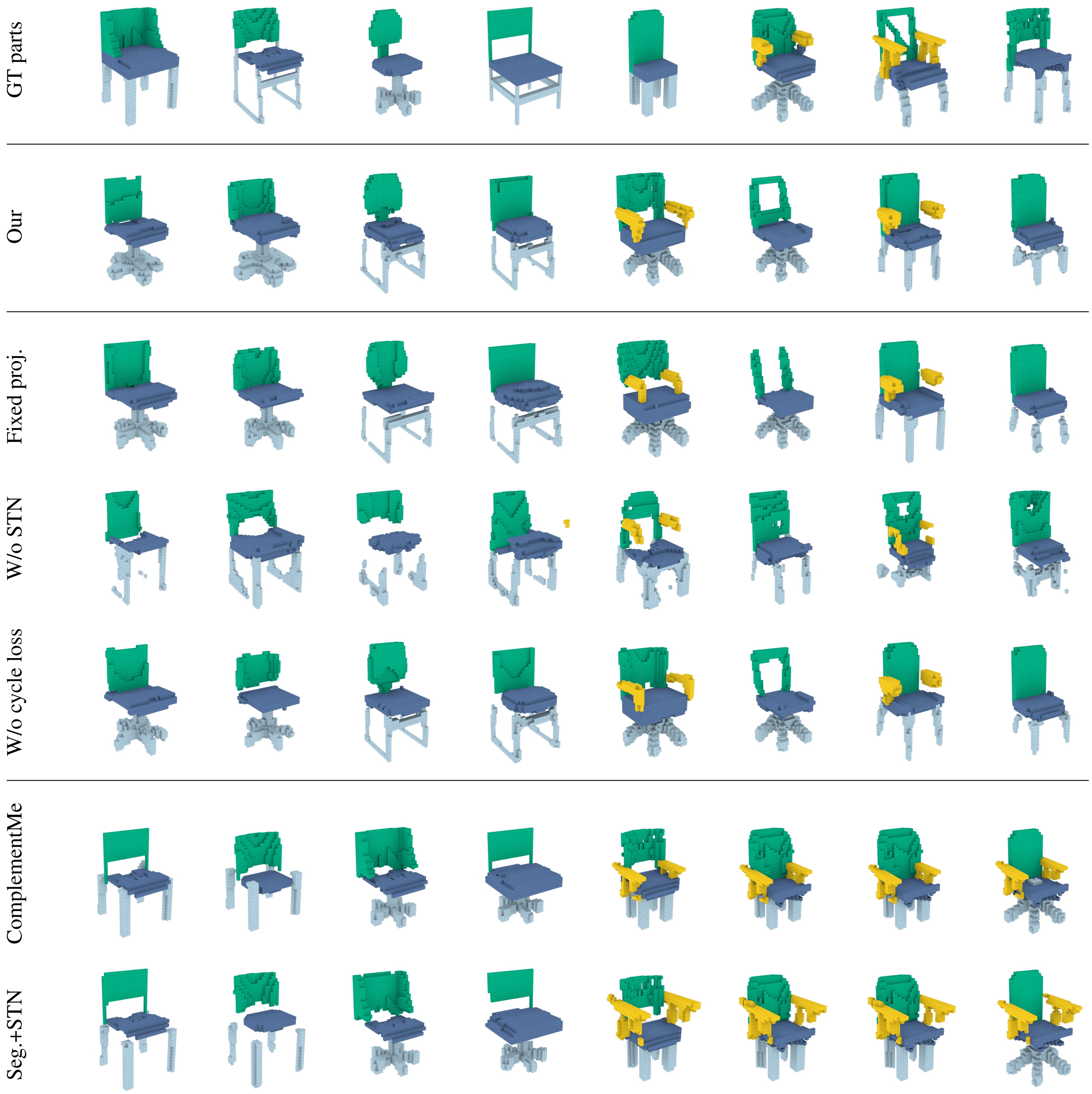


Figure 7: Ablation study: results of shape reconstruction from random parts. The top row shows input shapes with ground truth part labels. The following rows present the reconstruction result of the proposed method (Our), obtained with fixed projections (Fixed proj.), decoder without STN (W/o STN), and without cycle loss (W/o cycle loss).

In *Advances in Neural Information Processing Systems*, pages 82–90, 2016. 4, 8

[6] Zhijie Wu, Xiang Wang, Di Lin, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Sagnet: Structure-aware generative network for 3d-shape modeling. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2019)*, 38(4):91:1–91:14, 2019. 4

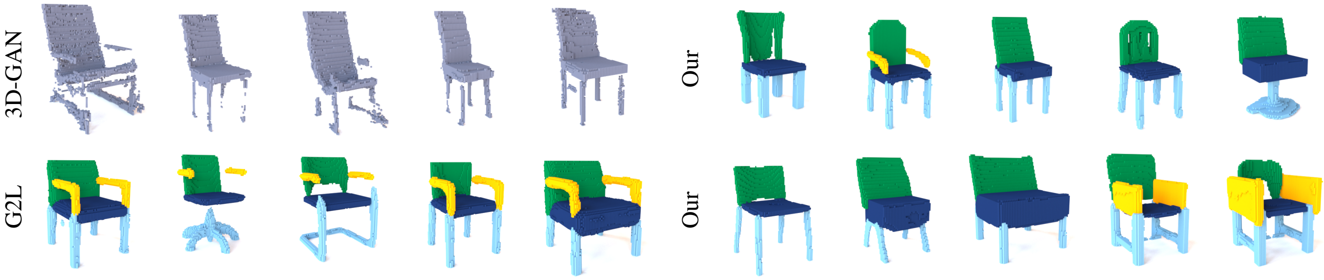


Figure 8: Shapes generated using 3D-GAN [5], G2L [4], and by random part assembly using our approach (using unsegmented shapes as input). Results were rendered using Mitsuba renderer `https://www.mitsuba-renderer.org/index.html`.

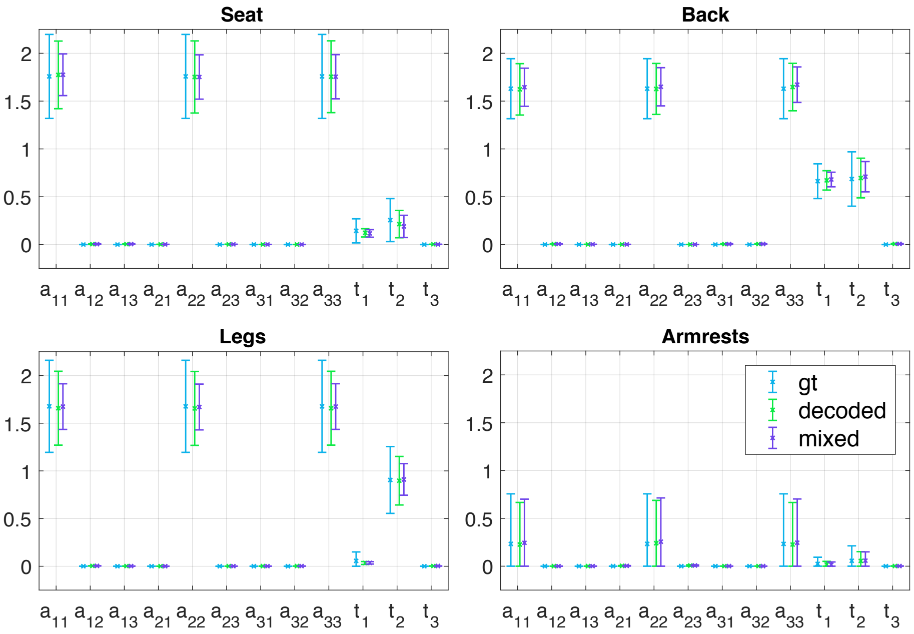


Figure 9: A comparison between the ground truth transformation parameters, and the parameters produced by our spatial transformer network, in the shape-from-part-assembly experiment.