

# PT2PC: Learning to Generate 3D Point Cloud Shapes from Part Tree Conditions

## Supplementary Material

Kaichun Mo<sup>1</sup>, He Wang<sup>1</sup>, Xinchun Yan<sup>2</sup>, and Leonidas Guibas<sup>1</sup>

<sup>1</sup> Stanford University  
<sup>2</sup> Uber ATG

This document provides supplementary materials accompanying the main paper, including

- More details on evaluation metrics;
- More details on the user study;
- More qualitative results of the generated point clouds;
- Mesh generation results;
- Discussion of failure cases and future works;
- Part Tree Visualization for Figure 4 of the Main Paper.

### A. More Details on Evaluation Metrics

In this section, we describe more details on the evaluation metrics: *coverage scores*, *diversity scores*, *Frechét Point-cloud Distance* and our proposed novel *HierInsSeg score*.

*Coverage Scores.* Conditioned on every part tree  $\mathcal{T}$ , the coverage score measures the average distance from each of the real shapes  $\mathbf{X}_{i,\text{real}} = \{X_i^j \mid P^j \in \mathcal{T}_{\text{leaf}}\}$  to the closest generated sample in  $\{\mathbf{X}_{j,\text{gen}}\}_j$ .

$$\text{Coverage Score}(\mathcal{T}) = \frac{1}{|\mathcal{X}_{\mathcal{T}}|} \sum_{X_{i,\text{real}} \in \mathcal{X}_{\mathcal{T}}} \left( \min_j \text{Dist}(X_{i,\text{real}}, X_{j,\text{gen}}) \right) \quad (1)$$

where  $\mathcal{X}_{\mathcal{T}}$  includes all real data samples  $\{\mathbf{X}_{i,\text{real}}\}_i$  that satisfies  $\mathcal{T}$ . We randomly generate 100 point cloud shapes  $\{\mathbf{X}_{j,\text{gen}}\}_{j=1}^{100}$ .

We introduce two variants of function  $\text{Dist}$  to measure the distance between two sets of part point clouds  $\mathbf{X}_{i_1}$  and  $\mathbf{X}_{i_2}$ .

$$\begin{aligned} \text{Dist}^{\text{part}}(\mathbf{X}_{i_1}, \mathbf{X}_{i_2}) &= \frac{1}{|\mathcal{T}_{\text{leaf}}|} \sum_{(j_1, j_2) \in \mathcal{M}} \text{EMD}(\mathbf{x}_{i_1}^{j_1}, \mathbf{x}_{i_2}^{j_2}) \\ \text{Dist}^{\text{shape}}(\mathbf{X}_{i_1}, \mathbf{X}_{i_2}) &= \text{EMD}(\text{DownSample}(\mathbf{X}_{i_1}), \text{DownSample}(\mathbf{X}_{i_2})) \end{aligned} \quad (2)$$

where  $\text{EMD}$  denotes the Earth Mover Distance [6,1] between two point clouds and  $\text{DownSample}$  is Furthest Point Sampling (FPS). Here,  $\mathcal{M}$  is the solution to a linear

sum assignment we compute over two sets of part point clouds  $\{\mathbf{x}_{i_1}^j \mid P^j \in \mathcal{T}_{\text{leaf}}\}$  and  $\{\mathbf{x}_{i_2}^j \mid P^j \in \mathcal{T}_{\text{leaf}}\}$  according to the part tree and part geometry.

We measure *part coverage score* and *shape coverage score* using  $\text{Dist}^{\text{part}}$  and  $\text{Dist}^{\text{shape}}$  respectively for every part tree condition  $\mathcal{T}$ , and finally average over all part trees to obtain the final coverage scores. The *shape coverage score* measures the holistic shape distance which is less structure-aware, while the *part coverage score* treats all parts equally and is more suitable to evaluate our part-tree conditioned generation task.

*Diversity Scores.* A good point cloud GAN should generate shapes with variations. We generate 10 point clouds for each part tree condition and compute diversity scores under both distance functions  $\text{Dist}^{\text{part}}$  and  $\text{Dist}^{\text{shape}}$ . Finally, we report the average *part diversity score* and *shape diversity score* across all part tree conditions.

$$\text{Diversity Score}(\mathcal{T}) = \frac{1}{100} \sum_{i,j=1}^{10} (\text{Dist}(X_{i,\text{gen}}, X_{j,\text{gen}})) \quad (3)$$

*Frechet Point-cloud Distance.* Shu *et al.* [7] introduced Frechet Point-cloud Distance (FPD) for evaluating the point cloud generation quality, inspired by the Frechet Inception Distance (FID) [2] commonly used for evaluating 2D image generation quality. A PointNet [4] is trained on ModelNet40 [8] for 3D shape classification and then FPD computes the real and fake feature distribution distance using the extracted point cloud global features from PointNet.

FPD jointly evaluates the generation quality, diversity and coverage. It is defined as

$$\text{Frechet Distance} = \|\mu_r - \mu_g\|^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}). \quad (4)$$

where  $\mu$  and  $\Sigma$  are the mean vector and the covariance matrix of the features for the real data distribution  $r$  and the generated one  $g$ . The notation  $\text{Tr}$  refers to the matrix trace.

As most of the part trees in PartNet have only one or few real shapes, we cannot easily compute a stable real data mean  $\mu_r$  and covariance matrix  $\Sigma_r$  for each part tree, which usually requires hundreds or thousands of data points to compute. Thus, we have to compute FPD over all part tree conditions by randomly sampling a part tree condition from the dataset and generating one shape point cloud conditioned on it. In this paper, we generate 10,000 shapes for each evaluation.

*HierInsSeg Score.* We propose a novel *HierInsSeg score*, which is a structural metric that measures how well the generated shape point clouds satisfy the symbolic part tree conditions. The *HierInsSeg* algorithm  $\text{Seg}(\mathbf{x})$  performs hierarchical part instance segmentation on the input shape point cloud  $\mathbf{x}$  and outputs a symbolic part tree depicting its part structure. Then we compute a tree-editing

distance between this part tree prediction and the part tree used as the generation condition. We perform a hierarchical Hungarian matching over the two symbolic part trees that matches according to the part semantics and the part subtree structures in a hierarchical top-down fashion. Any node mismatch in this procedure contributes to the tree difference score and the final tree-editing distance is computed by further divided by the total node count of the input part tree condition.

For each part tree, we conditionally generate 100 shape point clouds and compute the mean tree-editing distance. To get the *HierInsSeg* score, we simply average the mean tree-editing distances from all part trees.

Mo *et al.* [3] proposed a part instance segmentation method that takes as input a point cloud shape and outputs a variable number of disjoint part masks over the point cloud input, each of which represents a part instance. The method uses PointNet++ [5] as a backbone network that extracts per-point features over the input point cloud and then performs a 200-way classification for each point with a SoftMax layer that encourages every point belongs to one mask in the final outputs. Each of the 200 predicted masks is also associated with a score within  $[0, 1]$  indicating its existence. The existing and non-empty masks correspond to the final part segmentation. We refer to [3] for more details.

We propose our *HierInsSeg* algorithm  $\text{Seg}(\mathbf{x})$  by adapting [3] to a hierarchical setting. First, we compute the statistics over all training data to obtain the maximal number of parts for each part semantics in the canonical part semantic hierarchy. Then, we define a maximal instance-level part tree template  $\mathcal{T}^{\text{template}} = (\mathcal{T}_V^{\text{template}}, \mathcal{T}_E^{\text{template}})$  that covers all possible part trees in the training data. We adapt the same instance segmentation pipeline [3] but change the maximal number of output masks from 200 to  $|\mathcal{T}_V^{\text{template}}|$ . Finally, to make sure all children part masks sum up to the parent mask in the part tree template, we define

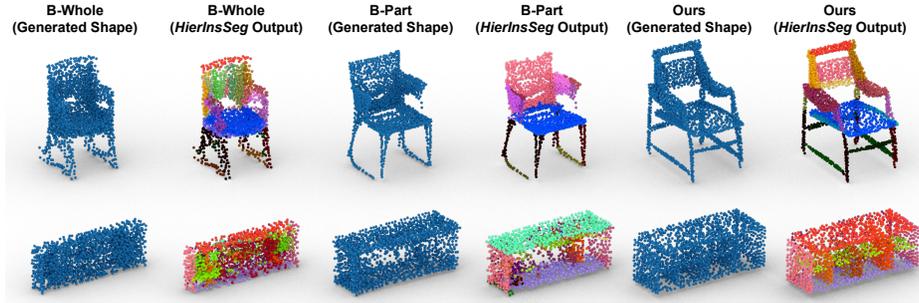
$$\mathbf{M}_j = \sum_{(j,k) \in \mathcal{T}_E^{\text{template}}} \mathbf{M}_k, \forall j \quad (5)$$

To implement this, for each parent part mask, we perform one SoftMax operation over all children part masks. The root node always has  $\mathbf{M}_{\text{root}} = \mathbf{1}$ .

In Table 2 of the main paper (the GT rows), we present the *HierInsSeg* scores operating on the real shape point clouds to provide a upper bound for the performance. In Figure 1, we also show qualitative results for performing the proposed hierarchical instance-level part segmentation over some example generated shapes. Both quantitative and qualitative results show that the proposed *HierInsSeg* algorithm is effective on judging if the generated shape point cloud satisfies the part tree condition.

## B. More Details on the User Study

We show our user study interface in Figure 4. We ask the users to rank three algorithms from three aspects: part structure, geometry, overall.



**Fig. 1. *HierInsSeg* Qualitative Results.** We show the input generated shape point clouds and the *HierInsSeg* results at the leaf level.

### C. More Qualitative Results

We present more qualitative results in Figure 5. Given the symbolic part trees as conditions, we show multiple diverse point clouds generated by our method.

### D. Mesh Generation Results

Since our method deforms a point cloud sampled from a unit cube mesh for each leaf-node part geometry, we naturally obtain the mesh generation results as well. Figure 2 shows some results. Since the goal of this work is primarily for point cloud generation, we do not explicitly optimize for the mesh generation results. However, we observe reasonable mesh generation results learned by our method.



**Fig. 2. Mesh Generation Results.** The top rows show the generated shape point clouds and the bottom rows show the corresponding generated mesh results.



**Fig. 3. failure cases.** The top row shows the real shapes and the bottom row presents our generated point clouds.

### E. Failure Cases and Future Works

Figure 3 presents common failure cases we observe. For the chair example, the back slats are not well aligned with each other and are unevenly distributed spatially. For the table example, the connecting parts between legs and surface extrude outside the table surface. In the cabinet example, the four drawers overlap with each other as the network does not assign clear roles for them. The lamp example has the disconnection problem between the rope and the base on the ceiling. All these cases indicate that future works should study how to better model part relationships and physical constraints.

### F. Part Tree Visualization for Figure 4 of the Main Paper

Figure 6 shows the eight part tree conditional inputs used for generating the point cloud shapes in Figure 4 of the main paper.

## References

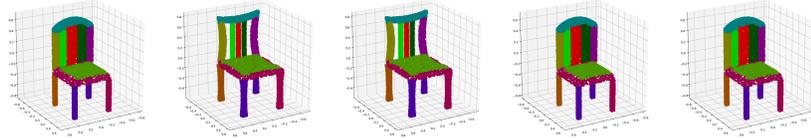
1. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 605–613 (2017)
2. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in neural information processing systems. pp. 6626–6637 (2017)
3. Mo, K., Zhu, S., Chang, A.X., Yi, L., Tripathi, S., Guibas, L.J., Su, H.: PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
4. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017)

5. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in neural information processing systems. pp. 5099–5108 (2017)
6. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover’s distance as a metric for image retrieval. *International journal of computer vision* **40**(2), 99–121 (2000)
7. Shu, D.W., Park, S.W., Kwon, J.: 3d point cloud generative adversarial network based on tree structured graph convolutions. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3859–3868 (2019)
8. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: CVPR. pp. 1912–1920 (2015)

Thank you for doing this user study! You will be asked to do 10 questions in this section (should be in 10 minutes). Thank you for help!

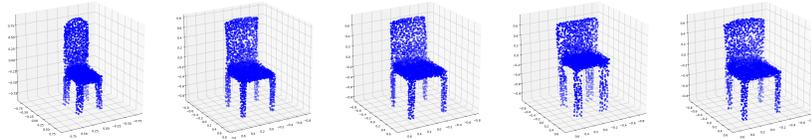
Current Progress: 0/10

Here are five ground-truth chairs (NO ORDER) satisfying a similar part-structure. Different parts are shown in different colors. Sometimes, the five examples can be identical. Don't penalize if the set of generated shapes contain plausible chair variations.

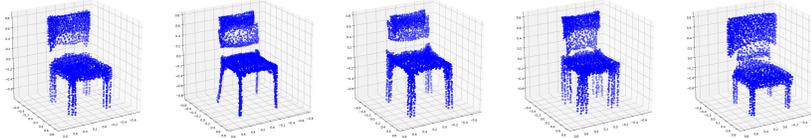


Please rank the following three sets of generated shape results A/B/C if they match the ground-truth shapes and if they are realistic (e.g. A>B>C means A is better than B and C is the worst). Each line shows five generated shapes (NO ORDER) from one algorithm.

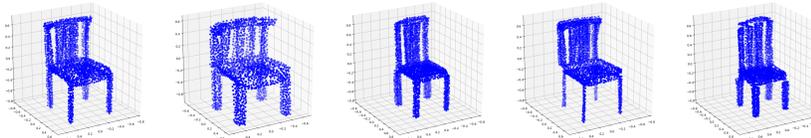
**Algorithm A: five generated shapes (NO ORDER)**



**Algorithm B: five generated shapes (NO ORDER)**



**Algorithm C: five generated shapes (NO ORDER)**



Please rank algorithms A/B/C under THREE criterion:

1) **Rank Shape-Part-Structure:** Please consider if the generated set of shapes contain clear part structures and if they match the ground-truth part-structure?

Not Answered! ↓

2) **Rank Shape-Geometry:** Regardless of the part/structures, how do you like the shape geometry? Lower your ranking if results contain visual artifacts, such as unevenly distributed points, disconnected parts, etc.

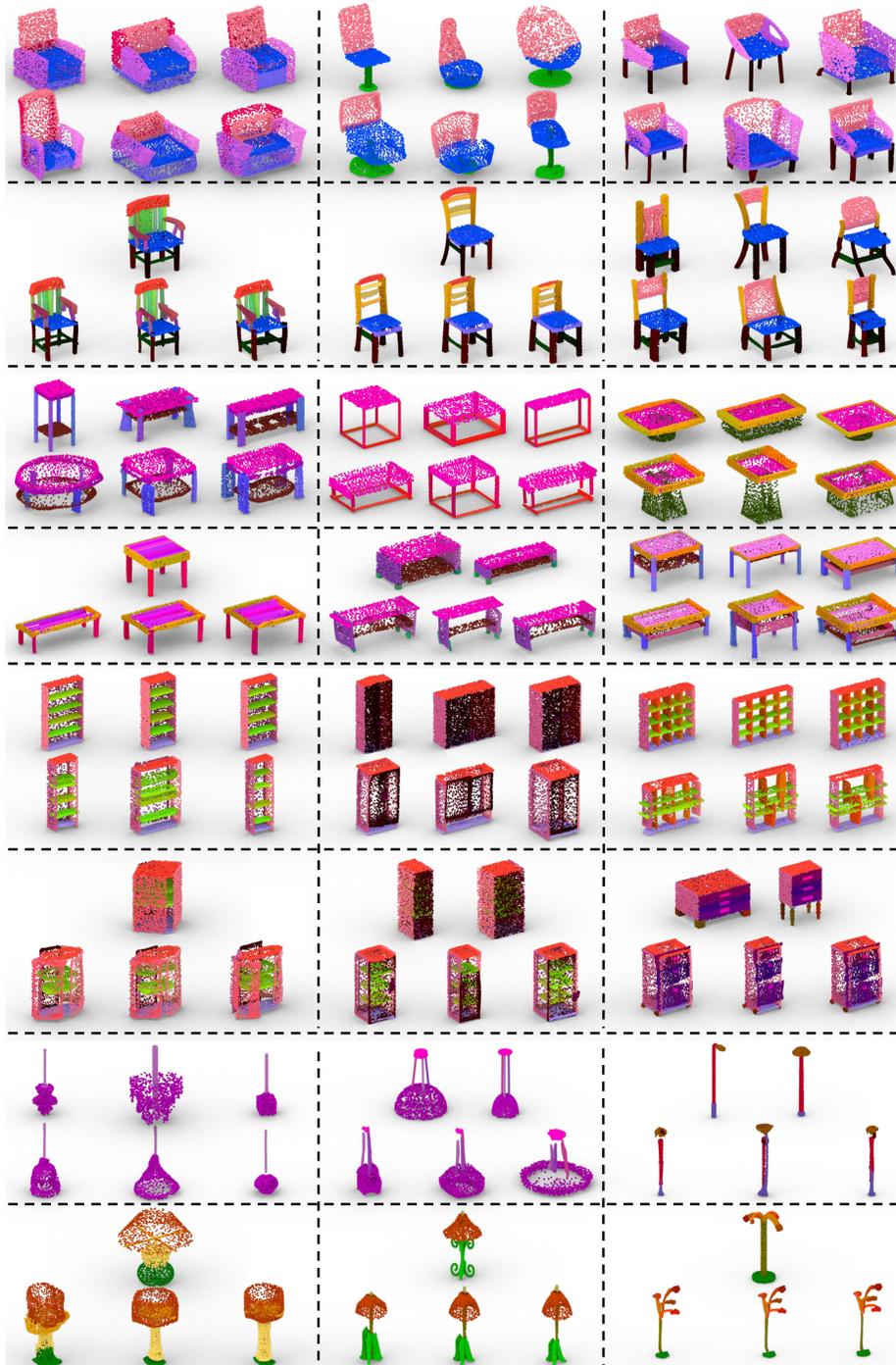
Not Answered! ↓

3) **Give an Overall Ranking:** Considering all the factors, give a final ranking for how the results agree to the kind of ground-truth chairs while being realistic.

Not Answered! ↓

Next

Fig. 4. User Study Interface.



**Fig. 5. Additional qualitative results.** We show six more results for each of the four categories. For each block, the top row shows the real shapes and the bottom row shows our generated results.

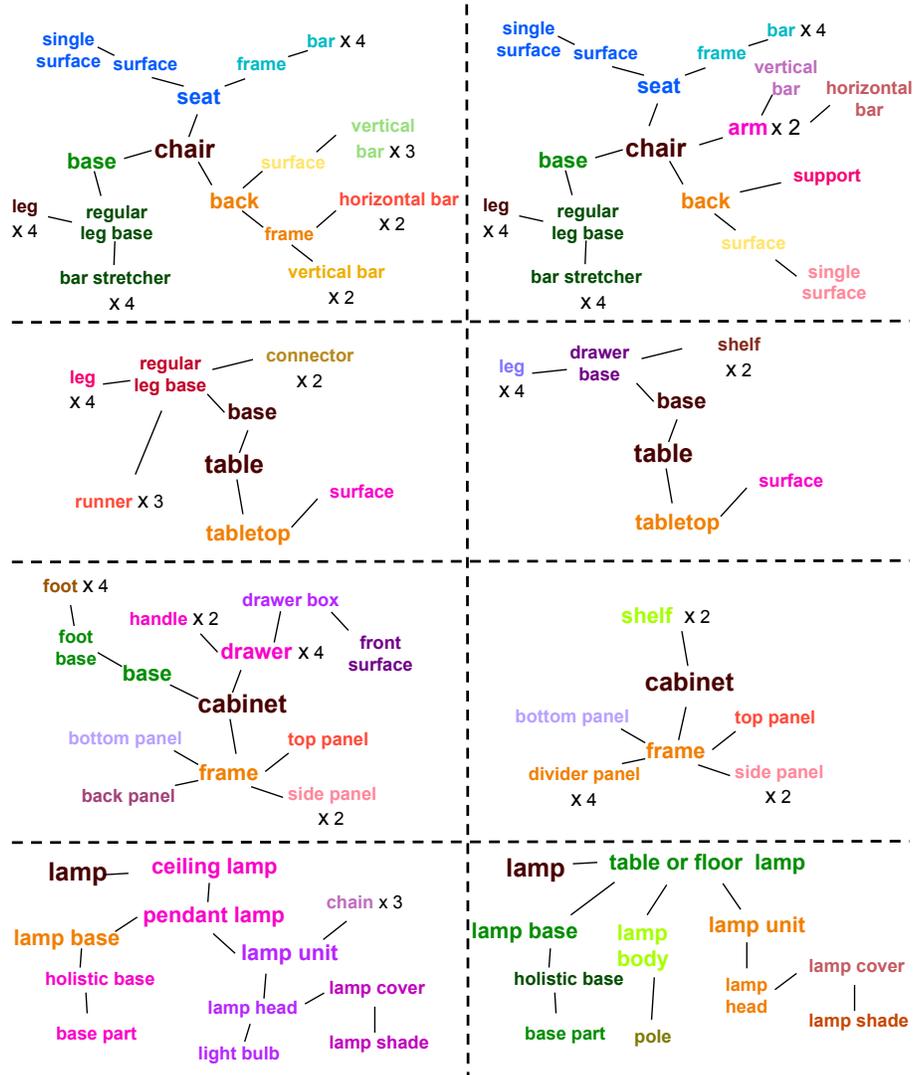


Fig. 6. Visualization for the Part Tree Conditions for Figure 4 of the main paper. Here we show the eight part tree conditional inputs used for generating the point cloud shapes in Figure 4 of the main paper.