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

Kaichun Mo[1], He Wang[1], Xinchen Yan[2], and Leonidas Guibas[1]

[1] Stanford University
[2] Uber ATG

Abstract. 3D generative shape modeling is a fundamental research area in computer vision and interactive computer graphics, with many real-world applications. This paper investigates the novel problem of generating a 3D point cloud geometry for a shape from a symbolic part tree representation. In order to learn such a conditional shape generation procedure in an end-to-end fashion, we propose a conditional GAN "part tree"-to-"point cloud" model (*PT2PC*) that disentangles the *structural* and *geometric* factors. The proposed model incorporates the part tree condition into the architecture design by passing messages *top-down* and *bottom-up* along the part tree hierarchy. Experimental results and user study demonstrate the strengths of our method in generating perceptually plausible and diverse 3D point clouds, given the part tree condition. We also propose a novel structural measure for evaluating if the generated shape point clouds satisfy the part tree conditions. Code and data are released on the webpage: https://cs.stanford.edu/~kaichun/pt2pc.

Keywords: part-tree to point-cloud, conditional generative adversarial network, part-based and structure-aware point cloud generation.

1 Introduction

3D shape generation is a central topic in computer vision and graphics. Recent works (*e.g.* [9,11,19]) have been focusing on generating the entire shape geometry without explicitly considering part semantics and shape structures. Such holistic shape generation pipelines, though successfully learning to model simple 3D shapes, usually have a difficult time modeling complicated shape structures and delicate shape parts. In computer-aided design (CAD), constructing a whole shape geometry from scratch is an extremely laborious and time-consuming task. If the designer only needs to give a sentence "a chair with 1 seat, 4 legs and a back with 3 bars" and the system can directly generate multiple shape candidates for her to select and edit from, it will save a big amount of time. Disentangling shape structure and geometry factors in shape generation also encourages more fine-grained and controllable 3D shape generation – thus supporting many real-world applications, including structure-conditioned shape design [41,39] and structure-aware shape re-synthesis [13].

In this paper, we formulate a new task of generating 3D point cloud shapes with *geometric* variations conditioned on *structural* shape descriptions. Figure 1

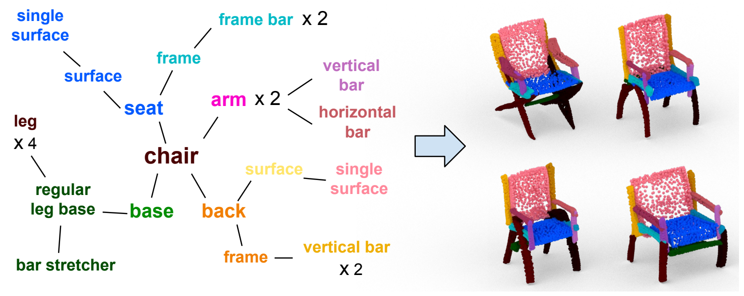


Fig. 1. We formulate the problem of "part tree"-to-"point cloud" (*PT2PC*) synthesis as a conditional generation task which takes in a symbolic part tree as condition and generates multiple diverse point clouds satisfying the structure defined by the part tree.

illustrates our task input and output with an example. More specifically, we represent each 3D shape as a hierarchy of parts, following PartNet [41], where each part node has an associated semantic label and the part hierarchy includes parts at different segmentation granularities. Abstracting away the concrete part geometry, the shape structure can be defined by a symbolic part tree with only part semantics and their relationships (Figure 1, left). Given such symbolic part tree conditions, we propose a conditional-GAN *PT2PC* to generate diverse 3D point cloud shapes that satisfy the structural conditions (Figure 1, right).

The symbolic part tree conditions are central to the architecture designs for both *PT2PC* generator and discriminator. Our generator first encodes the part tree template feature using semantic and structural information for each part node in a *bottom-up* fashion along the tree hierarchy. Then, given a random noise vector capturing the global geometry information at the root node, we recursively propagate such geometric information to each part node in a *top-down* fashion along the part tree. The final point clouds are generated by aggregating the point clouds decoded at each leaf node representing its corresponding fine-grained semantic part. Our discriminator first computes per-part features at the leaf level, propagates the information in a *bottom-up* fashion along the tree hierarchy until the root node and finally produces a score judging if the generated shape *geometry* looks realistic and the shape *structure* satisfies the input condition.

We evaluate the proposed model on four major semantic classes from the PartNet dataset. To justify the merits of our tree-structure architecture for both the generator and discriminator, we compare with two conditional GAN baselines. Both quantitative and qualitative results demonstrate clear advantages of our design in terms of global shape quality, part shape quality, and shape diversity, under both seen and unseen templates as the condition. Results on human evaluation agree with our observations in the experiments and further strengthen our claims. Additionally, we propose a novel hierarchical part instance segmentation method that is able to segment an input point cloud without any part labels into a symbolic part tree. This provides us a metric to evaluate how well our generated shape geometry satisfies the part tree conditions.

In summary, our contributions are:

- we formulate the novel task of part-tree conditioned point cloud generation;
- we propose a conditional GAN method, *PT2PC*, that generates realistic and diverse point cloud shapes given symbolic part tree conditions;
- we demonstrate superior performance both quantitatively and qualitatively under standard GAN metrics and a user study, comparing against two baseline conditional-GAN methods;
- we propose a novel point cloud structural evaluation metric for evaluating if the generated point clouds satisfy the part tree conditions.

2 Related Works

We review related works on 3D generative shape modeling, part-based shape modeling and structure-conditioned content generation.

3D Generative Shape Modeling. Reconstructing and synthesizing 3D shapes is a popular research topic in computer vision and graphics. Recently, tremendous progresses have been made in generating 3D voxel grids [9,18,66,67,70,74,49], point clouds [1,11,14,76,75], and surface meshes [53,21,19,37] using deep neural networks. Point clouds representation is a collection of unordered points irregularly distributed in the 3D space, which makes the minimax optimization very challenging [33,1]. Achlioptas *et al.* [1] proposed a latent-GAN approach that conducts minimax optimization on the shape feature space which outperforms the raw-GAN operating on the raw point clouds. To better capture the local geometric structure of point clouds, Valsesia *et al.* [60] proposed a graph-based generator that dynamically builds the graph based on distance in feature space. Shu *et al.* [52] proposed Tree-GAN with a tree-structured graph convolutional neural network as the generator. Recently, Wang *et al.* [63] proposed a new discriminator, PointNet-Mix, that improves the sampling uniformity of the generated point clouds. Unlike these shape point cloud GAN works that generate shapes without explicit part semantic and structural constraints, we learn to generate diverse point cloud shapes satisfying symbolic part tree conditions.

Part-based Shape Modeling. There is a line of research on understanding shapes by their semantic parts and structures. Previous works study part segmentation [7,30,78,28,45,79,64,41,10], box abstraction [59,85,43,54], shape template fitting [32,15,17,44], generating shapes by parts [29,35,55,62,69,57,39,68,16,51,34], or editing shape by parts [12,82,40]. We refer to the survey papers [72,38] for more related works. Shape parts have hierarchical structures [65,61,41]. Yi *et al.* [77] learns consistent part hierarchy from noisy online tagged shape parts. GRASS [35] propose binary part trees to generate novel shapes. A follow-up work [81] learns to segment shapes into the binary part hierarchy. PartNet [41] proposes a large-scale 3D model dataset with hierarchical part segmentation. Using PartNet, recent works such as StructureNet [39] and StructEdit [40] learns to generate and edit shapes explicitly following the pre-defined part hierarchy. We use the tree hierarchy defined in PartNet [41] and propose a new task *PT2PC* that learns to generate point cloud shapes given symbolic part tree conditions.

Conditional Content Generation. Understanding the 3D visual world, parsing the geometric and structural properties of 3D primitives (*e.g.* objects in the scene or parts of an object) and their relationships is at the core of computer vision [23,71,27,58]. Many works learn to synthesize high-quality images from text descriptions [31,48,47,83,80,36,56], semantic attributes [73,8], scene-graph representations [27,3], and rough object layouts [26,25,84,42]. There are also works to generate 3D content with certain input conditions. Chang *et al.* [5,4] learns to generate 3D scenes from text. Chen *et al.* [6] studied how to generate 3D voxel shapes from a sentence condition. StructEdit [40] learns to generate structural shape variations conditioned on an input source shape. Our work introduces a conditional Generative Adversarial Network that generates shape point clouds conditioned on an input symbolic part tree structure.

3 Method

In this work, we propose *PT2PC*, a conditional GAN (c-GAN) that learns a mapping from a given *symbolic part tree* $\mathcal{T}$ and a random noise vector $\mathbf{z}$ to a 3D shape point cloud $\mathbf{X}$ composed of part point clouds for the leaf nodes of the conditional part tree. We propose novel part-based conditional point cloud generator $G(\mathbf{z}, \mathcal{T})$ and discriminator $D(\mathbf{X}, \mathcal{T})$ conditioned on the symbolic part tree input $\mathcal{T}$. Different from holistic point cloud GANs [1,60,52] that produce a shape point cloud as a whole, our proposed *PT2PC* generate a hierarchy of part point clouds along with part semantics and shape structures.

3.1 Symbolic Part Tree Representation

We follow the semantic part hierarchy defined in PartNet [41]. Every PartNet shape instance (*e.g.* a chair) is annotated with a hierarchical part segmentation that provides both coarse-level parts (*e.g.* chair base, chair back) and parts at fine-grained levels (*e.g.* chair leg, chair back bar). Figure 2 shows the ground-truth part hierarchy of an exemplar chair.

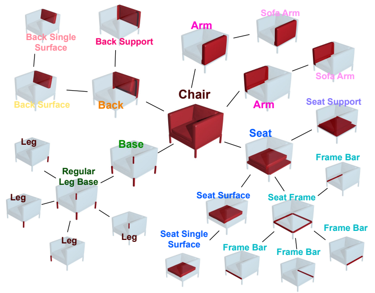


Fig. 2. An example PartNet hierarchical part segmentation.

A symbolic part tree $\mathcal{T}$ is defined as $\mathcal{T} = (\mathcal{T}_V, \mathcal{T}_E)$, where $\mathcal{T}_V = \{P^j | P^j = (\mathbf{s}^j, \mathbf{d}^j)\}_j$ represents a set of part instances and $\mathcal{T}_E$ represents an directed edge set of the part parent-children relationships $\mathcal{T}_E = \{(j, k)\}$. In $\mathcal{T}_V$, each part instance P^j is composed of two components: a semantic label $\mathbf{s}^j$ (*e.g.* chair seat, chair back), and a part instance identifier $\mathbf{d}^j$

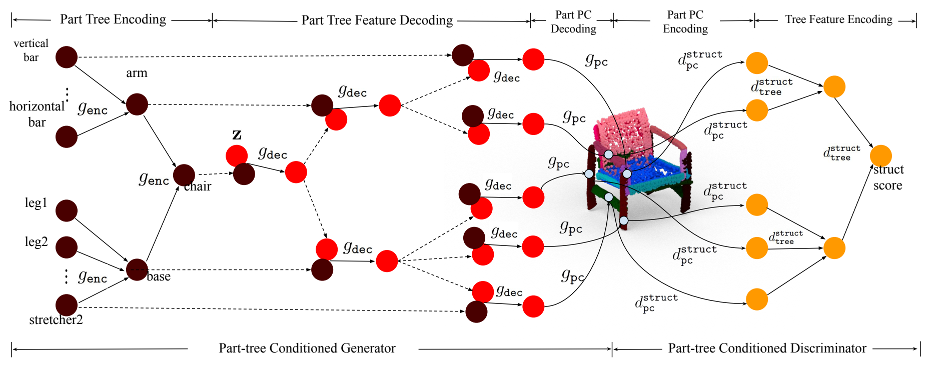


Fig. 3. Our c-GAN *PT2PC* architecture. Our part-tree conditioned generator first extracts the subtree features by traversing the input symbolic part tree in a *bottom-up* fashion, and then recursively decodes part features in a *top-down* way until leaf nodes, where part point clouds are finally generated. Our part-tree conditioned geometry discriminator recursively consumes the generated part tree with leaf node geometry in a *bottom-up* fashion to generate the final score. The solid arrow indicates a network module while a dashed arrow means to copy the content. As defined in Sec 3, the brown, red and orange nodes represent the encoded symbolic part feature $\mathbf{t}$, the decoded part geometry feature $\mathbf{f}$ and the encoded part geometry feature $\mathbf{h}$ respectively.

(*e.g.* the first leg, the second leg), both of which are represented as one-hot vectors. The set of part semantic labels are pre-defined in PartNet and consistent within one object category. In $\mathcal{T}_E$, each edge (j,k) indicates P^j is the parent node of P^k. The set $C^j = \{k|(j,k) \in \mathcal{T}_E\}$ defines all children part instances of a node P^j. We denote a special part node $P^{\texttt{root}}$ to be the root node of the part tree $\mathcal{T}$, with the semantic label $\mathbf{s}^{\texttt{root}}$ and the instance identifier $\mathbf{d}^{\texttt{root}}$. The leaf node of the symbolic part tree has no children, namely, $\mathcal{T}_{\texttt{leaf}} = \left\{P^j \mid |C^j| = 0\right\} \subsetneq \mathcal{T}_V$.

3.2 Part-tree Conditioned Generator

Our conditional generator $G(\mathbf{z}, \mathcal{T})$ takes a random Gaussian variable $\mathbf{z} \sim \mathcal{N}(\cdot|\mu = 0, \sigma = 1)$ and a symbolic part tree condition $\mathcal{T} = (\mathcal{T}_V, \mathcal{T}_E)$ as inputs and outputs a set of part point clouds $\mathbf{X} = \{\mathbf{x}^j \mid P^j \in \mathcal{T}_{\texttt{leaf}}\}$ where $\mathbf{x}^j \in \mathbb{R}^{M\times 3}$ is a part point cloud in the shape space representing the leaf node part P^j. Namely,

$$\mathbf{X} = G(\mathbf{z}, \mathcal{T}) \tag{1}$$

The generator is composed of three network modules: a symbolic part tree encoder $G_{\texttt{enc}}$, a part tree feature decoder $G_{\texttt{dec}}$ and a part point cloud decoder $G_{\texttt{pc}}$. First, the symbolic part tree encoder $G_{\texttt{enc}}$ embeds the nodes of $\mathcal{T}$ into compact features $\mathbf{t}^j$ hierarchically from the leaf nodes to the root node for every node P^j. Then, taking in both the random variable $\mathbf{z}$ and the hierarchy of symbolic part features $\{\mathbf{t}^j\}_j$, the part tree feature decoder $G_{\texttt{dec}}$ hierarchically decodes the part features in the top-down manner, from the root node to the leaf nodes, and finally produces part feature $\mathbf{f}^j$ for every leaf node $P^j \in \mathcal{T}_{\mathtt{leaf}}$. Finally, the part point cloud decoder $G_{\mathtt{pc}}$ transforms the leaf node features $\left\{\mathbf{f}^j \mid P^j \in \mathcal{T}_{\mathtt{leaf}}\right\}$ into 3D part point clouds $\left\{\mathbf{x}^j \mid P^j \in \mathcal{T}_{\mathtt{leaf}}\right\}$ in the shape space.

At each step of the part feature decoding, the parent node needs to know the global structural context in order to propagate coherent signals to all of its children so that the generated part point clouds can form a valid shape in a compatible way. This is the reason why we introduce the symbolic part tree encoder $G_{\mathtt{enc}}$ as a *bottom-up* module to summarize the part tree structural context for each decoding step. Our part tree decoder $G_{\mathtt{dec}}$ is then conditioned on the symbolic structural context and recursively propagates the random noise $\mathbf{z}$ from the root node to the leaf nodes in a *top-down* fashion.

Symbolic part tree encoder G_{enc}. For a given symbolic part tree $\mathcal{T}$, we encode the nodes of the part tree starting from the leaf nodes and propagate the messages to the parent node of the encoded nodes until the root node gets encoded. The message propagation is performed in a *bottom-up* fashion. As shown in Eq. 2, each node P^j takes the node feature $\mathbf{t}^k$, the semantic label $\mathbf{s}^k$ and the part instance identifier $\mathbf{d}^k$ from all its children $\left\{P^k | k \in C^j\right\}$, aggregates the information and computes its node feature $\mathbf{t}^j$ using a learned function g_{enc}. Then, it further propagates a message to its parent node. We initialize $\mathbf{t}^j = 0$ for every leaf node $P^j \in \mathcal{T}_{\mathtt{leaf}}$.

$$\begin{aligned} \mathbf{t}^j &= \mathbf{0}, && \forall P^j \in \mathcal{T}_{\mathtt{leaf}} \\ \mathbf{t}^j &= g_{\mathtt{enc}}\left(\left\{[\mathbf{t}^k; \mathbf{s}^k; \mathbf{d}^k] \mid k \in C^j\right\}\right), && \forall P^j \in \mathcal{T}_V - \mathcal{T}_{\mathtt{leaf}} \end{aligned} \tag{2}$$

where $[\cdot;\cdot]$ means a concatenation of the inputs. $g_{\mathtt{enc}}$ is implemented as a small PointNet[45], treating each children-node feature as a high dimensional point, to enforce the permutation invariance between children nodes. We first use a fully-connected layer to embed each $[\mathbf{t}^k; \mathbf{s}^k; \mathbf{d}^k]$ into a 256-dim feature, then perform a max-pooling over $K = |C^j|$ features over all children nodes to obtain an aggregated feature, and finally push the aggregated feature through another fully-connected layer to obtain the final parent node feature $\mathbf{t}^j$. We use leaky `ReLU` as the activation functions in our fully-connected layers.

Part tree feature decoder G_{dec}. Taking in the random variable $\mathbf{z}$ and encoded node features $\{\mathbf{t}^j\}_j$, we hierarchically decode the part features $\{\mathbf{f}^j\}_j$ from the root node to the leaf nodes in a *top-down* fashion along the given part tree structure $\mathcal{T}$. As shown in Eq. 3, for every part P^j, we learn a shared function g_{dec} transforming the concatenation of its own features $(\mathbf{t}^j, \mathbf{s}^j, \mathbf{d}^j)$ and the decoded feature $\mathbf{f}^p$ from its parent node P^p into part feature $\mathbf{f}^j$. For the root node, we use random noise $\mathbf{z}$ to replace parent node feature.

$$\begin{aligned} \mathbf{f}^{\mathtt{root}} &= g_{\mathtt{dec}}\left([\mathbf{z}; \mathbf{t}^{\mathtt{root}}; \mathbf{s}^{\mathtt{root}}; \mathbf{d}^{\mathtt{root}}]\right), \\ \mathbf{f}^j &= g_{\mathtt{dec}}\left([\mathbf{f}^p; \mathbf{t}^j; \mathbf{s}^j; \mathbf{d}^j]\right), && \forall (p,j) \in \mathcal{T}_E \end{aligned} \tag{3}$$

We implement $g_{\mathtt{dec}}$ as a two-layer MLP with leaky `ReLU` as the activation functions. The output feature size is 256.

Part point cloud decoder G_{pc}. Given the part features of all the leaf nodes $\left\{\mathbf{f}^j \mid P^j \in \mathcal{T}_{\texttt{leaf}}\right\}$, our point cloud decoder $G_{\texttt{pc}}$ transforms each individual feature $\mathbf{f}^j$ into a 3D part point cloud $\mathbf{x}^j$ in the shape space for every $P^j \in \mathcal{T}_{\texttt{leaf}}$, as shown in Eq. 4. To get the final shape point cloud, we down-sample the union of all part point clouds $\left\{\mathbf{x}^j \mid P^j \in \mathcal{T}_{\texttt{leaf}}\right\}$. We generate the same number M points for all the parts.

$$\begin{aligned} \mathbf{x}^j &= g_{\texttt{pc}}(\mathbf{f}^j), \forall P^j \in \mathcal{T}_{\texttt{leaf}} \\ \mathbf{x} &= \texttt{DownSample}\left(\cup\left\{\mathbf{x}^j \mid P^j \in \mathcal{T}_{\texttt{leaf}}\right\}\right) \end{aligned} \tag{4}$$

$g_{\texttt{pc}}$ is designed to deform a fixed surface point cloud of a unit cube $\mathbf{x}_{\texttt{cube}}$ into our target part point cloud based on its input $\mathbf{f}$, inspired by the shape decoder introduced in Groueix *et al.* [20]. We uniformly sample a 1000-size point cloud from the surface of a unit cube to form $\mathbf{x}_{\texttt{cube}} \in \mathbb{R}^{1000\times 3}$. Then, for each point in $\mathbf{x}_{\texttt{cube}}$, we concatenate its XYZ coordinate with the feature $\mathbf{f}$, push it through a $MLP(256+3, 1024, 1024, 3)$ using leaky `ReLU`, and finally obtain an XYZ coordinate for a point on our target point cloud. Finally, we use Furthest Point Sampling (FPS) for our `Downsample` operation to obtain shape point cloud $\mathbf{x}$.

Compared to existing works [1,60,52] that generate shape point clouds as a whole, the key difference here is that our point cloud decoder generates part point clouds for every leaf node in the part tree $\mathcal{T}$ separately, but in a manner aware of the inter-part structure and relationships. Another big advantage is that we get the semantic label of each generated part point cloud. Furthermore, we observe that the holistic point cloud generators usually suffer from non-uniform point distribution. The generators tend to allocate way more points to bulky parts (*e.g.*, chair back and chair seat) while only generating sparse points for small parts with thin geometry (*e.g.*, chair wheel, chair back bar). Since our $G_{\texttt{pc}}$ generates the same number of points for each part and then performs global down-sampling, we can generate shape point clouds with fine structures and appropriate point density for all the parts.

3.3 Part-tree Conditioned Discriminator

Our conditional discriminator $D(\mathbf{X}, \mathcal{T})$ receives a generated sample or a true data sample, composed of a set of part point clouds $\mathbf{X} = \left\{\mathbf{x}^j \in \mathbb{R}^{M\times 3} | P^j \in \mathcal{T}_{\texttt{leaf}}\right\}$, and outputs a scalar $y \in \mathbb{R}$ based on the tree condition $\mathcal{T}$. Following the WGAN-gp [2,22], D is learned to be a 1-Lipschitz function of $\mathbf{X}$ and its output y depicts the realness of the sample.

Since the input $\mathbf{X}$ always contains part point clouds for every leaf node part instances in the symbolic part tree $\mathcal{T}$, our discriminator mainly focus on judging the geometry of each part point clouds along with the whole shape point clouds assembled from the parts. This is to say, the discriminator should tell if each part point cloud is realistic and plausible regarding its part semantics; in addition, the discriminator needs to look at the spatial arrangement of the part point clouds, judge whether it follows a realistic structure specified by the part tree $\mathcal{T}$, *e.g.* connected parts need to contact each other and some parts may exhibit certain kind of symmetry; finally, the discriminator should judge whether the generated part point clouds form a realistic shape point cloud.

To address the requirements above, our discriminator leverages two modules: a structure-aware part point cloud discriminator $D^{\texttt{struct}}(\mathbf{X}, \mathcal{T})$, and a holistic shape point cloud discriminator $D^{\texttt{whole}}(\mathbf{x})$, where $\mathbf{x} = \texttt{DownSample}(\cup\mathbf{X})$. $D^{\texttt{struct}}(\mathbf{X}, \mathcal{T})$ takes as input the part tree condition $\mathcal{T}$ and the generated set of part point clouds $\mathbf{X}$ and outputs a scalar $y^{\texttt{struct}} \in \mathbb{R}$ regarding the tree-conditioned generation quality. $D^{\texttt{whole}}(\mathbf{x}) \in \mathbb{R}$ only takes the down-sampled shape point cloud $\mathbf{x}$ as input and outputs a scalar $y^{\texttt{whole}}$ regarding the unconditioned shape quality. As shown in Eq.5, the final output of our discriminator D is the sum of the two discriminators.

$$\begin{aligned} y &= y^{\texttt{struct}} + y^{\texttt{whole}} \\ y^{\texttt{struct}} &= D^{\texttt{struct}}(\mathbf{X}, \mathcal{T}) \\ y^{\texttt{whole}} &= D^{\texttt{whole}}(\mathbf{x}) \end{aligned} \tag{5}$$

For the structure-aware part point cloud discriminator $D^{\texttt{struct}}(\mathbf{X}, \mathcal{T})$, we constitute it using three network components: a part point cloud encoder $D^{\texttt{struct}}_{\texttt{pc}}$, a tree-based feature encoder $D^{\texttt{struct}}_{\texttt{tree}}$, and a scoring network $D^{\texttt{struct}}_{\texttt{score}}$. First, the point cloud encoder $D^{\texttt{struct}}_{\texttt{pc}}$ encodes the part point cloud $\mathbf{x}^j$ into a part feature $\mathbf{h}^j$ for each leaf node $P^j \in \mathcal{T}_{\texttt{leaf}}$. Then, taking in the part features at leaf level $\left\{\mathbf{h}^j \mid P^j \in \mathcal{T}_{\texttt{leaf}}\right\}$, the tree-based feature encoder $D^{\texttt{struct}}_{\texttt{tree}}$ recursively propagates the part features $\mathbf{h}$ along with the part semantics $\mathbf{s}$ to the parent nodes starting from the leaf nodes and finally reaching the root node, in a *bottom-up* fashion. Finally, a scoring function $D^{\texttt{struct}}_{\texttt{score}}$ outputs a score $y^{\texttt{struct}} \in \mathbb{R}$ for the shape generation quality. For the holistic shape point cloud discriminator $D^{\texttt{whole}}$, it is simply composed of a PointNet encoder $D^{\texttt{whole}}_{\texttt{pc}}$ and a scoring network $D^{\texttt{whole}}_{\texttt{score}}$ which outputs a scalar $y^{\texttt{whole}} \in \mathbb{R}$.

Point cloud encoder D^{struct}_{pc} and D^{whole}_{pc}. Both $D^{\texttt{struct}}_{\texttt{pc}}$ and $D^{\texttt{whole}}_{\texttt{pc}}$ use vanilla PointNet [45] architecture without spatial transformer layers or batch normalization layers. For $D^{\texttt{struct}}_{\texttt{pc}}$, we learn a function $d^{\texttt{struct}}_{\texttt{pc}}$ to extract a part geometry feature $\mathbf{h}^j$ for each part point cloud $\mathbf{x}^j$.

$$\mathbf{h}^j = d^{\texttt{struct}}_{\texttt{pc}}(\mathbf{x}^j), \forall P^j \in \mathcal{T}_{\texttt{leaf}} \tag{6}$$

$d^{\texttt{struct}}_{\texttt{pc}}$ is implemented as a four-layer $MLP(3, 64, 128, 128, 1024)$ to process each point individually followed by a max-pooling. Similarly, $D^{\texttt{whole}}_{\texttt{pc}}$ takes a shape point cloud $\mathbf{x}$ as input and outputs a global shape feature $\mathbf{h}^{\texttt{shape}}$.

$$\mathbf{h}^{\texttt{shape}} = D^{\texttt{whole}}_{\texttt{pc}}(\mathbf{x}) \tag{7}$$

Tree feature encoder D^{struct}_{tree}. Similar to the symbolic part tree encoder $G_{\texttt{dec}}$ in the generator, $D^{\texttt{struct}}_{\texttt{tree}}$ learns an aggregation function $d^{\texttt{struct}}_{\texttt{tree}}$ that transforms features from children nodes into parent node features, as shown in Eq. 8. By leveraging the tree structure specified by $\mathcal{T}$ in its architecture, the module enforces the structure-awareness of $D^{\texttt{struct}}$. In a *bottom-up* fashion, the features propagate from the leaf level finally to the root yielding $\mathbf{h}^{\texttt{root}}$, according to Eq.8.

$$\mathbf{h}^j = d^{\texttt{struct}}_{\texttt{tree}}\left(\left\{[\mathbf{h}^k;\mathbf{s}^k] \mid k \in C^j\right\}\right), \forall P^j \in \mathcal{T}_V - \mathcal{T}_{\texttt{leaf}} \tag{8}$$

To implement $D^{\texttt{struct}}_{\texttt{tree}}$, we extract a latent 256-dim feature after applying a fully-connected layer over each input $[\mathbf{h}^k;\mathbf{s}^k]$, perform max-pooling over all children nodes and finally push it through another fully-connected layer to obtain $\mathbf{h}^j$. We use the leaky `ReLU` activation functions for both layers.

Note that the key difference between $D^{\texttt{struct}}_{\texttt{tree}}$ and $G_{\texttt{enc}}$ is that $D^{\texttt{struct}}_{\texttt{tree}}$ no longer requires the part instance identifiers $\mathbf{d}^k$ since the children part features $\{\mathbf{h}^k\}_k$ for each parent node P^j already encode the part geometry information that are naturally different even for part instances of the same part semantics.

Scoring functions D^{part}_{score} and D^{whole}_{score}. After obtaining the structure-aware root feature $\mathbf{h}^{\texttt{root}}$ and the holistic PointNet feature $\mathbf{h}^{\texttt{shape}}$, we compute

$$\begin{aligned} y^{\texttt{struct}} &= D^{\texttt{struct}}_{\texttt{score}}\left(\mathbf{h}^{\texttt{root}}\right) \\ y^{\texttt{whole}} &= D^{\texttt{whole}}_{\texttt{score}}\left(\mathbf{h}^{\texttt{shape}}\right) \end{aligned} \tag{9}$$

Both scoring functions are implemented as a simple fully-connected layer with no activation function.

3.4 Training

We follow WGAN-gp [2,22] for training our *PT2PC* conditional generator $G(\cdot,\mathcal{T})$ and discriminator $D(\cdot,\mathcal{T})$. The objective function is defined in Eq. 10.

$$\mathcal{L} = \mathbb{E}_{\mathbf{z}\sim\mathcal{Z}}[D(G(\mathbf{z},\mathcal{T}),\mathcal{T})] - \mathbb{E}_{\mathbf{X}\sim\mathcal{R}}[D(\mathbf{X},\mathcal{T})] + \lambda_{gp}\mathbb{E}_{\hat{\mathbf{X}}}\left[(\|\nabla_{\hat{\mathbf{X}}}D(\hat{\mathbf{X}},\mathcal{T})\|_2 - 1)^2\right] \tag{10}$$

where we interpolate each pair of corresponding part point clouds from a real set $\mathbf{X}_{\texttt{real}} = \left\{\mathbf{x}^j_{\texttt{real}} | P^j \in \mathcal{T}_{\texttt{leaf}}\right\}$ and a fake set $\mathbf{X}_{\texttt{fake}} = \left\{\mathbf{x}^j_{\texttt{fake}} | P^j \in \mathcal{T}_{\texttt{leaf}}\right\}$ to get $\hat{\mathbf{X}} = \left\{\hat{\mathbf{x}}^j | P^j \in \mathcal{T}_{\texttt{leaf}}\right\}$, as shown in below:

$$\hat{\mathbf{x}}^j = \alpha \cdot \mathbf{x}^j_{\texttt{real}} + (1-\alpha)\cdot \mathbf{x}^j_{\texttt{fake}}, \forall P^j \in \mathcal{T}_{\texttt{leaf}}, \tag{11}$$

where $\alpha \in (0,1)$ is a random interpolation coefficient always remaining same for all parts. We iteratively train the generator and discriminator with $n_{critic} = 10$. We choose $\lambda_{gp} = 1$, $N = 2,048$ and $M = 1,000$ in our experiments. And, we assume the maximal children per parent node to be 10.

4 Experiments

We evaluate our proposed *PT2PC* on the PartNet [41] dataset and compare to two baseline c-GAN methods. We show superior performance on all standard point cloud GAN metrics. Besides, we propose a new structural metric evaluating how well the generated point clouds satisfy the input part tree conditions, based on a novel hierarchical instance-level shape part segmentation algorithm. We also conduct a user study which confirms our strengths over baseline methods.

4.1 Dataset

We use the PartNet [41] dataset as our main testbed. PartNet contains fine-grained and hierarchical instance-level semantic part annotations including 573,585 part instances over 26,671 3D models covering 24 categories. In this paper, we use the four largest categories that contain diverse part structures: chairs, tables, cabinets and lamps. Following StructureNet [39], we assume there are at maximum 10 children for every parent node and remove the shapes containing unlabeled parts for the canonical sets of part semantics in PartNet [41]. Table 1 summarizes data statistics and the train/test splits. We split by part trees with a ratio 75%:25%. See Table 1 for more details. We observe that most part trees (*e.g.* 1,787 out of 2,197 for chairs) have only one real data point in PartNet, which posts challenges to generate shapes with geometry variations.

Table 1. Dataset Statistics. We summarize the number of shapes and symbolic part trees in the train and test splits for each object category.

Category	#Shapes			#Part Trees		
	Total	Train	Test	Total	Train	Test
Chair	4871	3848	1023	2197	1648	549
Table	5099	4146	953	1267	925	342
Cabinet	846	606	240	619	470	149
Lamp	802	569	233	302	224	78

4.2 Baselines

We compare to two vanilla versions of conditional GAN methods as baselines.

– **Whole-shape Vanilla c-GAN (`B-Whole`):** the method uses a Multiple-layer Perception (MLP) $G_{\texttt{baseline}}(\mathbf{z}, \mathcal{T})$ as the generator and the holistic shape point cloud discriminator $D^{\texttt{whole}}_{\texttt{baseline}}(\mathbf{x}, \mathcal{T})$;
– **Part-based Vanilla c-GAN (`B-Part`):** the method uses exactly the same proposed generator $G(\mathbf{z}, \mathcal{T})$ as in our method and a holistic shape point cloud discriminator $D^{\texttt{whole}}_{\texttt{baseline}}(\mathbf{x}, \mathcal{T})$.

One can think `B-Part` as an ablated version of our proposed method, without the structural discriminator $D^{\texttt{struct}}(\mathbf{X})$. The `B-Whole` method further removes our part-based generator and generates whole shape point clouds in one shot, similar to previous works [1,60,52]. We implement $D^{\texttt{whole}}_{\texttt{baseline}}(\mathbf{x}, \mathcal{T})$ similar to $D^{\texttt{whole}}(\mathbf{x})$ used as part of our discriminator. It uses a vanilla PointNet [45] to extract global geometry features for input point clouds. Additionally, to make it be aware of the input part tree condition $\mathcal{T}$, we re-purpose the proposed part tree feature encoder network $G_{\texttt{enc}}$ in our generator to recursively compute a root node feature summarizing the entire part tree structural information. We make $D^{\texttt{whole}}_{\texttt{baseline}}(\mathbf{x}, \mathcal{T})$

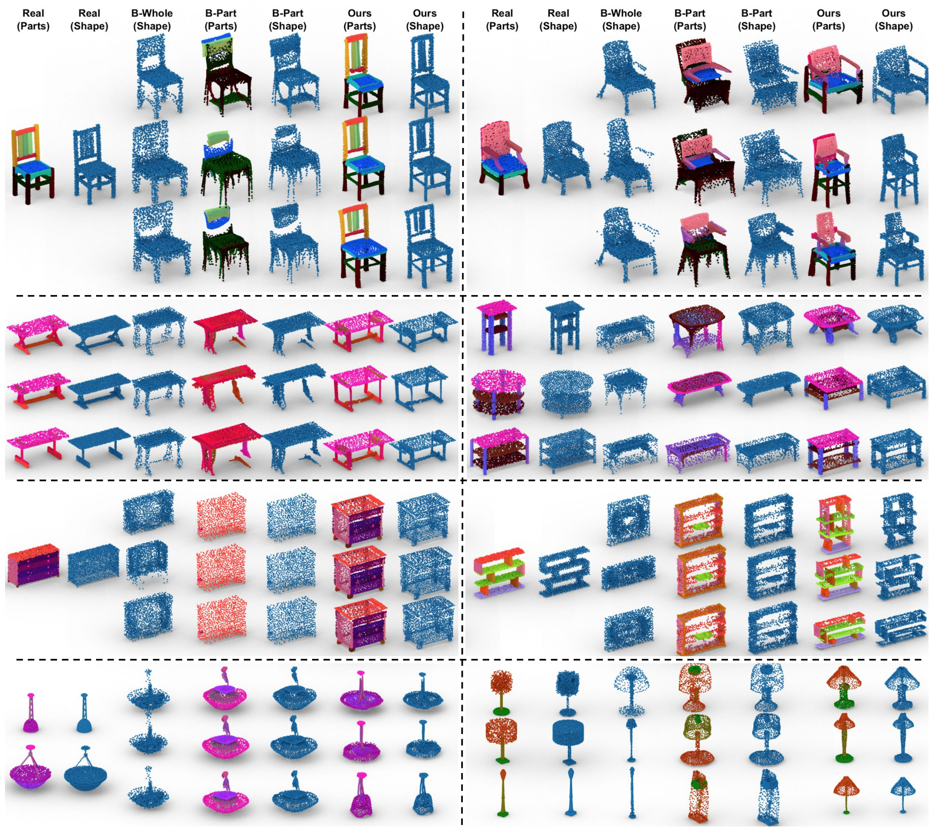


Fig. 4. Qualitative Comparisons. We show two examples for each of the four categories: chair, table, cabinet and lamp. The leftmost two columns show the real examples illustrating the conditional part tree input (see Figure 11 for the input part tree visualization). We show three random real examples unless there are only one or two in the dataset. For our method and `B-Part` we show both the generated part point clouds with colors and the down-sampled shape point clouds, to fairly compare with `B-Whole` that only produces shape point clouds.

conditional on the extracted root node feature. For $G_{\texttt{baseline}}(\mathbf{z}, \mathcal{T})$, we follow Achlioptas *et al.* [1] and design a five-layer MLP with sizes 512, 512, 512, 1024, 2048×3 that finally produces a point cloud of size 2048×3. We use leaky `ReLU` as activation functions except for the final output layer. We also condition $G_{\texttt{baseline}}(\mathbf{z}, \mathcal{T})$ on the root feature extracted from the template feature encoder.

4.3 Metrics

We report standard point cloud GAN metrics, including coverage, diversity [1], and Frechét Point-cloud Distance (FPD) [52]. Note that coverage and diversity originally measure the distance between shape point clouds, which is, more or less, structure-unaware. We introduce two structure-aware metrics, *part coverage* and *part diversity* adopting the original ones by evaluating the average distance between corresponding parts of the two shapes. In addition, we devise a novel perceptual structure-aware metric *HierInsSeg* that measures the part tree edit distance leveraging deep neural networks.

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

$$\text{Coverage Score}(\mathcal{T}) = \frac{1}{|\mathcal{X}_\mathcal{T}|} \sum_{X_{i,\mathtt{real}} \in \mathcal{X}_\mathcal{T}} \left(\min_j \mathtt{Dist}\left(X_{i,\mathtt{real}}, X_{j,\mathtt{gen}}\right) \right) \tag{12}$$

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

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

$$\begin{aligned} \mathtt{Dist}^{\mathtt{part}}\left(\mathbf{X}_{i_1}, \mathbf{X}_{i_2}\right) &= \frac{1}{|\mathcal{T}_{\mathtt{leaf}}|} \sum_{(j_1, j_2) \in \mathcal{M}} \mathtt{EMD}\left(\mathbf{x}_{i_1}^{j_1}, \mathbf{x}_{i_2}^{j_2}\right) \\ \mathtt{Dist}^{\mathtt{shape}}\left(\mathbf{X}_{i_1}, \mathbf{X}_{i_2}\right) &= \mathtt{EMD}\left(\mathtt{DownSample}(\mathbf{X}_{i_1}), \mathtt{DownSample}(\mathbf{X}_{i_2})\right) \end{aligned} \tag{13}$$

where $\mathtt{EMD}$ denotes the Earth Mover Distance [50,11] between two point clouds and $\mathtt{DownSample}$ 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 $\left\{\mathbf{x}_{i_1}^j \mid P^j \in \mathcal{T}_{\mathtt{leaf}}\right\}$ and $\left\{\mathbf{x}_{i_2}^j \mid P^j \in \mathcal{T}_{\mathtt{leaf}}\right\}$ according to the part tree and part geometry.

We measure *part coverage score* and *shape coverage score* using $\mathtt{Dist}^{\mathtt{part}}$ and $\mathtt{Dist}^{\mathtt{shape}}$ 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 $\mathtt{Dist}^{\mathtt{part}}$ and $\mathtt{Dist}^{\mathtt{shape}}$. Finally, we report the average *part diversity score* and *shape diversity score* across all part tree conditions.

$$\text{Diversity Score}(\mathcal{T}) = \frac{1}{100} \sum_{i,j=1}^{10} \left(\mathtt{Dist}\left(X_{i,\mathtt{gen}}, X_{j,\mathtt{gen}}\right) \right) \tag{14}$$

Frechét Point-cloud Distance. Shu *et al.* [52] introduced Frechét Point-cloud Distance (FPD) for evaluating the point cloud generation quality, inspired by the Frechét Inception Distance (FID) [24] commonly used for evaluating 2D image generation quality. A PointNet [45] is trained on ModelNet40 [70] 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

$$\text{Frechet Distance} = ||\mu_r - \mu_g||^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2\,(\Sigma_r \Sigma_g)^{1/2}). \tag{15}$$

where μ and Σ are the mean vector and the covariance matrix of the features for the real data distribution r and the generated one g. The notation 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 μ_r and covariance matrix Σ_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 `Seg(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.* [41] 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++ [46] 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 [41] for more details.

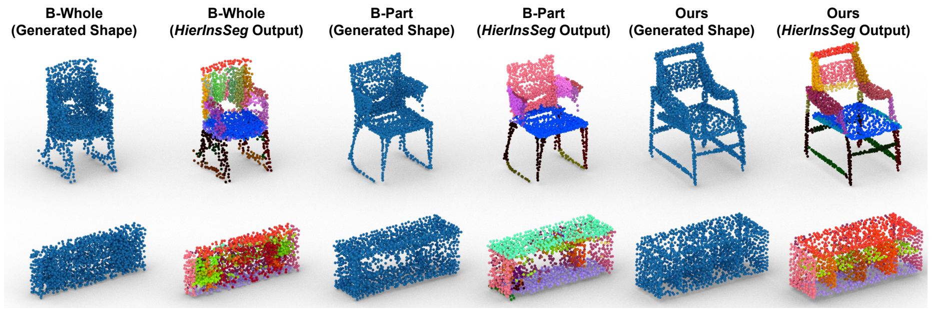


Fig. 5. *HierInsSeg* Qualitative Results. We show the input generated shape point clouds and the *HierInsSeg* results at the leaf level.

We propose our *HierInsSeg* algorithm $\texttt{Seg}(\mathbf{x})$ by adapting [41] 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 $\mathcal{T}^{\texttt{template}} = (\mathcal{T}_V^{\texttt{template}}, \mathcal{T}_E^{\texttt{template}})$ that covers all possible part trees in the training data. We adapt the same instance segmentation pipeline [41] but change the maximal number of output masks from 200 to $\left|\mathcal{T}_V^{\texttt{template}}\right|$. Finally, to make sure all children part masks sum up to the parent mask in the part tree template, we define

$$\mathbf{M}_j = \sum_{(j,k)\in\mathcal{T}_E^{\texttt{template}}} \mathbf{M}_k, \forall j \tag{16}$$

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

In Table 2 (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 5, 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.

4.4 Results and Analysis

We train our proposed *PT2PC* method and the two vanilla c-GAN baselines on the training splits of the four PartNet categories. The part trees in the test splits are unseen during the training time. Table 2 summarizes the quantitative evaluations. Our *HierInsSeg* scores are always the best as we explicitly generate part point clouds and hence render clearer part structures. Moreover, we win most of the FPD scores, showing that our method can generate realistic point cloud shapes. Finally, we find that our part-based generative model usually provides higher shape diversity as a result of part compositionality.

Table 2. Quantitative Evaluations. We report the quantitative metric scores for our *PT2PC* framework and the two vanilla c-GAN baselines. S and P are short for Shape and Part. Cov, Div, HIS are short for *coverage* score, *diversity* score and *HierInsSeg* score. Since the baseline `B-Whole` does not predict part point clouds, so *part coverage score* and *part diversity score* cannot be defined. We also report the ground-truth *HierInsSeg* scores for each category. The last two rows show the ablation study on chair, where Ours-W is ours without $D^{\texttt{whole}}$.

	Method	Train						Test					
		S-Cov↓	P-Cov↓	S-Div↑	P-Div↑	FPD↓	HIS↓	S-Cov	P-Cov	S-Div	P-Div	FPD	HIS
Chair	B-Whole	**0.13**	–	0.14	–	7.32	0.57	**0.13**	–	0.13	–	10.88	0.57
	B-Part	0.14	0.41	0.14	0.06	7.17	0.58	0.15	0.41	0.14	0.06	11.10	0.58
	Ours	**0.13**	**0.06**	**0.18**	**0.07**	**6.64**	**0.48**	0.14	**0.07**	**0.18**	**0.07**	**10.69**	**0.48**
	GT						0.30						0.31
Table	B-Whole	**0.19**	–	0.14	–	13.02	1.04	**0.21**	–	0.14	–	20.63	1.02
	B-Part	0.20	0.60	0.15	**0.09**	6.45	1.02	**0.21**	0.60	0.15	**0.09**	16.92	0.99
	Ours	0.21	**0.11**	**0.18**	**0.09**	**5.58**	**0.93**	0.23	**0.17**	**0.17**	**0.09**	**15.33**	**0.89**
	GT						0.62						0.64
Cabinet	B-Whole	0.15	–	0.09	–	16.38	0.90	**0.17**	–	**0.08**	–	22.90	0.86
	B-Part	0.30	0.84	0.03	0.01	**3.25**	0.64	0.43	0.84	0.03	0.01	24.29	0.81
	Ours	**0.13**	**0.08**	**0.13**	**0.02**	4.13	**0.52**	0.24	**0.18**	0.05	**0.02**	**17.73**	**0.57**
	GT						0.32						0.35
Lamp	B-Whole	0.38	–	0.08	–	17.87	1.00	**0.38**	–	0.09	–	86.96	0.96
	B-Part	**0.28**	0.73	0.09	0.03	6.52	0.78	0.43	0.70	0.09	0.03	94.66	0.88
	Ours	0.32	**0.04**	**0.11**	**0.05**	**5.71**	**0.68**	0.41	**0.19**	**0.12**	**0.05**	**80.55**	**0.83**
	GT						0.51						0.57
Chair	Ours-W	0.14	0.07	**0.22**	**0.08**	10.60	0.51	0.15	**0.07**	**0.21**	**0.08**	13.52	0.49
Abla.	Ours	**0.13**	**0.06**	0.18	0.07	**6.64**	**0.48**	**0.14**	**0.07**	0.18	0.07	**10.69**	**0.48**

Figure 4 shows qualitative comparisons of our method to the two baseline methods. One can clearly observe that `B-Whole` produces holistically reasonable shape geometry but with unclear part structures, which explains why it achieves decent shape coverage scores but fails to match our method under FPD and HIS. For `B-Part`, it fails severely for chair, table and cabinet examples that it does not assign clear roles for the parts and the generated part point clouds are overlaid with each other, which explains the high part coverage scores in Table 2. Obviously, our method generates shapes with clearer part structures and boundaries. We also see a reasonable amount of generation diversity even for part trees with only one real data in PartNet, thanks to the knowledge sharing among similar part tree and sub-tree structures when training a unified and conditional network. We also conduct an ablation study on chairs where we remove the holistic discriminator $D^{\texttt{whole}}$.

4.5 User Study

Although we provide both Euclidean metrics (*i.e.* coverage and diversity scores) and perceptual metrics (*i.e.* FPD and the proposed *HierInsSeg* scores) for evaluating generation quality in Table 2, the true measure of success is human judgement of the generated shapes. For this reason we perform a user study to evaluate the generation quality on chair class. For each trial, we show users a part tree

Table 3. User Study Results on Chair Generation. Here we show the average ranking of the three methods. The ranking ranges from 1 (the best) to 3 (the worst). The results on train templates are calculated based on 267 trials while the results on test templates are from the rest 269 trials.

	Train			Test		
	Structure	Geometry	Overall	Structure	Geometry	Overall
B-Whole	2.39	2.07	2.22	2.40	2.10	2.21
B-Part	2.33	2.41	2.38	2.36	2.47	2.46
Ours	**1.29**	**1.51**	**1.40**	**1.24**	**1.43**	**1.33**

as the condition, 5 ground truth shapes as references, and 5 randomly generated shape point clouds for each of the three methods. We ask users to rank the methods regarding the following three aspects: 1) structure similarity to the given part tree; 2) geometry plausibility; 3) overall generation quality. For fair comparison, we randomize the order between the methods in all trials and only show the shape point clouds without part labels. In total, we collected 536 valid records from 54 users. In Table 3, we report the average ranking of the three methods. Our method significantly outperforms the other two baseline methods on all of the three aspects and on both train and test templates. Please refer to Appendix Sec. A for the user interface and more details.

4.6 Decoupling Geometry and Structure for PC-GAN

Our proposed *PT2PC* framework enables disentanglement of shape *structure* and *geometry* generation factors. We demonstrate the capability of exploring structure-preserving geometry variation and geometry-preserving structure variation using our method. Conditioned on the same symbolic part tree, our network is able to generate shape point clouds with *geometric* variations by simply changing the Gaussian random noise $\mathbf{z}$. On the other hand, if we fix the same noise $\mathbf{z}$, conditioned on different input part trees, we observe that *PT2PC* is able to produce *geometrically* similar but *structurally* different shapes. Figure 6 shows the generated shape point clouds $\{\mathbf{x}_{i,j} = G(\mathbf{z}_i, \mathcal{T}_j)\}$ from a set of Gaussian noises $\{\mathbf{z}_i\}_i$ and a set of part trees $\{\mathcal{T}_j\}_j$. Each row shows shape structural interpolation results while sharing similar shape geometry, and every column presents geometric interpolation results conditioned on the same part tree structure.

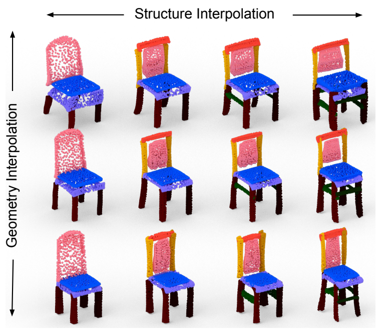


Fig. 6. Our approach enables disentanglement of *geometry* and *structure* factors in point cloud generation. Each row shares the same Gaussian noise $\mathbf{z}$ and every column is conditioned on the same part tree input.

5 Conclusion

We have proposed *PT2PC*, a conditional generative adversarial network (c-GAN) that generates point cloud shapes given a symbolic part-tree condition. The part tree input specifies a hierarchy of semantic part instances with their parent-children relationships. Extensive experiments and user study show our superior performance compared to two baseline c-GAN methods. We also propose a novel metric *HierInsSeg* to evaluate if the generated shape point clouds satisfy the part tree conditions. Future works may study incorporating more part relationships and extrapolating our method to unseen categories.

Acknowledgments

This research was supported by a Vannevar Bush Faculty Fellowship, grants from the Samsung GRO program and the Stanford SAIL Toyota Research Center, and gifts from Autodesk and Adobe.

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: ICML. pp. 40–49 (2018)
2. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein gan. arXiv preprint arXiv:1701.07875 (2017)
3. Ashual, O., Wolf, L.: Specifying object attributes and relations in interactive scene generation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4561–4569 (2019)
4. Chang, A., Monroe, W., Savva, M., Potts, C., Manning, C.D.: Text to 3d scene generation with rich lexical grounding. arXiv preprint arXiv:1505.06289 (2015)
5. Chang, A., Savva, M., Manning, C.D.: Learning spatial knowledge for text to 3d scene generation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 2028–2038 (2014)
6. Chen, K., Choy, C.B., Savva, M., Chang, A.X., Funkhouser, T., Savarese, S.: Text2shape: Generating shapes from natural language by learning joint embeddings. In: Asian Conference on Computer Vision. pp. 100–116. Springer (2018)
7. Chen, X., Golovinskiy, A., Funkhouser, T.: A benchmark for 3d mesh segmentation. Acm transactions on graphics (tog) **28**(3), 1–12 (2009)
8. Choi, Y., Choi, M., Kim, M., Ha, J.W., Kim, S., Choo, J.: Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8789–8797 (2018)
9. Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In: ECCV. pp. 628–644. Springer (2016)
10. Deng, B., Genova, K., Yazdani, S., Bouaziz, S., Hinton, G., Tagliasacchi, A.: Cvxnets: Learnable convex decomposition. arXiv preprint arXiv:1909.05736 (2019)
11. 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)

12. Fish, N., Averkiou, M., Van Kaick, O., Sorkine-Hornung, O., Cohen-Or, D., Mitra, N.J.: Meta-representation of shape families. ACM Transactions on Graphics (TOG) **33**(4), 1–11 (2014)
13. Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S., Dobkin, D.: Modeling by example. ACM transactions on graphics (TOG) **23**(3), 652–663 (2004)
14. Gadelha, M., Wang, R., Maji, S.: Multiresolution tree networks for 3d point cloud processing. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 103–118 (2018)
15. Ganapathi-Subramanian, V., Diamanti, O., Pirk, S., Tang, C., Niessner, M., Guibas, L.: Parsing geometry using structure-aware shape templates. In: 2018 International Conference on 3D Vision (3DV). pp. 672–681. IEEE (2018)
16. Gao, L., Yang, J., Wu, T., Yuan, Y.J., Fu, H., Lai, Y.K., Zhang, H.: Sdm-net: Deep generative network for structured deformable mesh. ACM Transactions on Graphics (TOG) **38**(6), 1–15 (2019)
17. Genova, K., Cole, F., Vlasic, D., Sarna, A., Freeman, W.T., Funkhouser, T.: Learning shape templates with structured implicit functions. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 7154–7164 (2019)
18. Girdhar, R., Fouhey, D.F., Rodriguez, M., Gupta, A.: Learning a predictable and generative vector representation for objects. In: ECCV. pp. 484–499. Springer (2016)
19. Gkioxari, G., Malik, J., Johnson, J.: Mesh r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9785–9795 (2019)
20. Groueix, T., Fisher, M., Kim, V.G., Russell, B.C., Aubry, M.: Atlasnet: A papier-mâché approach to learning 3d surface generation. In: CVPR (2018)
21. Groueix, T., Fisher, M., Kim, V.G., Russell, B.C., Aubry, M.: A papier-mâché approach to learning 3d surface generation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 216–224 (2018)
22. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: Advances in neural information processing systems. pp. 5767–5777 (2017)
23. Gupta, A., Efros, A.A., Hebert, M.: Blocks world revisited: Image understanding using qualitative geometry and mechanics. In: European Conference on Computer Vision. pp. 482–496. Springer (2010)
24. 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)
25. Hong, S., Yan, X., Huang, T.S., Lee, H.: Learning hierarchical semantic image manipulation through structured representations. In: Advances in Neural Information Processing Systems. pp. 2708–2718 (2018)
26. Hong, S., Yang, D., Choi, J., Lee, H.: Inferring semantic layout for hierarchical text-to-image synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7986–7994 (2018)
27. Johnson, J., Gupta, A., Fei-Fei, L.: Image generation from scene graphs. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1219–1228 (2018)
28. Kalogerakis, E., Averkiou, M., Maji, S., Chaudhuri, S.: 3d shape segmentation with projective convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3779–3788 (2017)

29. Kalogerakis, E., Chaudhuri, S., Koller, D., Koltun, V.: A probabilistic model for component-based shape synthesis. ACM Transactions on Graphics (TOG) **31**(4), 1–11 (2012)
30. Kalogerakis, E., Hertzmann, A., Singh, K.: Learning 3d mesh segmentation and labeling (2010)
31. Karacan, L., Akata, Z., Erdem, A., Erdem, E.: Learning to generate images of outdoor scenes from attributes and semantic layouts. arXiv preprint arXiv:1612.00215 (2016)
32. Kim, V.G., Li, W., Mitra, N.J., Chaudhuri, S., DiVerdi, S., Funkhouser, T.: Learning part-based templates from large collections of 3d shapes. ACM Transactions on Graphics (TOG) **32**(4), 1–12 (2013)
33. Li, C.L., Zaheer, M., Zhang, Y., Poczos, B., Salakhutdinov, R.: Point cloud gan. arXiv preprint arXiv:1810.05795 (2018)
34. Li, J., Niu, C., Xu, K.: Learning part generation and assembly for structure-aware shape synthesis. In: AAAI (2020)
35. Li, J., Xu, K., Chaudhuri, S., Yumer, E., Zhang, H., Guibas, L.: Grass: Generative recursive autoencoders for shape structures. ACM Transactions on Graphics (TOG) **36**(4), 1–14 (2017)
36. Li, W., Zhang, P., Zhang, L., Huang, Q., He, X., Lyu, S., Gao, J.: Object-driven text-to-image synthesis via adversarial training. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12174–12182 (2019)
37. Liao, Y., Donne, S., Geiger, A.: Deep marching cubes: Learning explicit surface representations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2916–2925 (2018)
38. Mitra, N.J., Wand, M., Zhang, H., Cohen-Or, D., Kim, V., Huang, Q.X.: Structure-aware shape processing. In: ACM SIGGRAPH 2014 Courses. pp. 1–21 (2014)
39. Mo, K., Guerrero, P., Yi, L., Su, H., Wonka, P., Mitra, N., Guibas, L.: Structurenet: Hierarchical graph networks for 3d shape generation. ACM Transactions on Graphics (TOG), Siggraph Asia 2019 **38**(6), Article 242 (2019)
40. Mo, K., Guerrero, P., Yi, L., Su, H., Wonka, P., Mitra, N., Guibas, L.: StructEdit: Learning structural shape variations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020)
41. 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)
42. Mo, S., Cho, M., Shin, J.: Instagan: Instance-aware image-to-image translation. arXiv preprint arXiv:1812.10889 (2018)
43. Niu, C., Li, J., Xu, K.: Im2struct: Recovering 3d shape structure from a single rgb image. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4521–4529 (2018)
44. Paschalidou, D., Ulusoy, A.O., Geiger, A.: Superquadrics revisited: Learning 3d shape parsing beyond cuboids. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 10344–10353 (2019)
45. 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)
46. 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)

47. Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H.: Generative adversarial text to image synthesis. arXiv preprint arXiv:1605.05396 (2016)
48. Reed, S.E., Akata, Z., Mohan, S., Tenka, S., Schiele, B., Lee, H.: Learning what and where to draw. In: Advances in neural information processing systems. pp. 217–225 (2016)
49. Riegler, G., Osman Ulusoy, A., Geiger, A.: Octnet: Learning deep 3d representations at high resolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3577–3586 (2017)
50. 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)
51. Schor, N., Katzir, O., Zhang, H., Cohen-Or, D.: Componet: Learning to generate the unseen by part synthesis and composition. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 8759–8768 (2019)
52. 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)
53. Sinha, A., Unmesh, A., Huang, Q., Ramani, K.: Surfnet: Generating 3d shape surfaces using deep residual networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6040–6049 (2017)
54. Sun, C.Y., Zou, Q.F., Tong, X., Liu, Y.: Learning adaptive hierarchical cuboid abstractions of 3d shape collections. ACM Transactions on Graphics (TOG) **38**(6), 1–13 (2019)
55. Sung, M., Su, H., Kim, V.G., Chaudhuri, S., Guibas, L.: Complementme: weakly-supervised component suggestions for 3d modeling. ACM Transactions on Graphics (TOG) **36**(6), 1–12 (2017)
56. Tan, F., Feng, S., Ordonez, V.: Text2scene: Generating compositional scenes from textual descriptions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6710–6719 (2019)
57. Tian, Y., Luo, A., Sun, X., Ellis, K., Freeman, W.T., Tenenbaum, J.B., Wu, J.: Learning to infer and execute 3d shape programs. arXiv preprint arXiv:1901.02875 (2019)
58. Tulsiani, S., Gupta, S., Fouhey, D.F., Efros, A.A., Malik, J.: Factoring shape, pose, and layout from the 2d image of a 3d scene. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 302–310 (2018)
59. Tulsiani, S., Su, H., Guibas, L.J., Efros, A.A., Malik, J.: Learning shape abstractions by assembling volumetric primitives. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2635–2643 (2017)
60. Valsesia, D., Fracastoro, G., Magli, E.: Learning localized generative models for 3d point clouds via graph convolution (2018)
61. Van Kaick, O., Xu, K., Zhang, H., Wang, Y., Sun, S., Shamir, A., Cohen-Or, D.: Co-hierarchical analysis of shape structures. ACM Transactions on Graphics (TOG) **32**(4), 1–10 (2013)
62. Wang, H., Schor, N., Hu, R., Huang, H., Cohen-Or, D., Huang, H.: Global-to-local generative model for 3d shapes. ACM Transactions on Graphics (TOG) **37**(6), 1–10 (2018)
63. Wang, H., Jiang, Z., Yi, L., Mo, K., Su, H., Guibas, L.J.: Rethinking sampling in 3d point cloud generative adversarial networks. arXiv preprint arXiv:2006.07029 (2020)
64. Wang, P., Gan, Y., Shui, P., Yu, F., Zhang, Y., Chen, S., Sun, Z.: 3d shape segmentation via shape fully convolutional networks. Computers & Graphics **70**, 128–139 (2018)

65. Wang, Y., Xu, K., Li, J., Zhang, H., Shamir, A., Liu, L., Cheng, Z., Xiong, Y.: Symmetry hierarchy of man-made objects. In: Computer graphics forum. pp. 287–296. No. 2, Wiley Online Library (2011)
66. Wu, J., Wang, Y., Xue, T., Sun, X., Freeman, B., Tenenbaum, J.: Marrnet: 3d shape reconstruction via 2.5 d sketches. In: NIPS. pp. 540–550 (2017)
67. Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In: NIPS. pp. 82–90 (2016)
68. Wu, R., Zhuang, Y., Xu, K., Zhang, H., Chen, B.: Pq-net: A generative part seq2seq network for 3d shapes. arXiv preprint arXiv:1911.10949 (2019)
69. Wu, Z., Wang, X., Lin, D., Lischinski, D., Cohen-Or, D., Huang, H.: Sagnet: Structure-aware generative network for 3d-shape modeling. ACM Transactions on Graphics (TOG) **38**(4), 1–14 (2019)
70. 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)
71. Xu, D., Zhu, Y., Choy, C.B., Fei-Fei, L.: Scene graph generation by iterative message passing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5410–5419 (2017)
72. Xu, K., Kim, V.G., Huang, Q., Mitra, N., Kalogerakis, E.: Data-driven shape analysis and processing. In: SIGGRAPH ASIA 2016 Courses. pp. 1–38 (2016)
73. Yan, X., Yang, J., Sohn, K., Lee, H.: Attribute2image: Conditional image generation from visual attributes. In: European Conference on Computer Vision. pp. 776–791. Springer (2016)
74. Yan, X., Yang, J., Yumer, E., Guo, Y., Lee, H.: Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In: NIPS. pp. 1696–1704 (2016)
75. Yang, G., Huang, X., Hao, Z., Liu, M.Y., Belongie, S., Hariharan, B.: Pointflow: 3d point cloud generation with continuous normalizing flows. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4541–4550 (2019)
76. Yang, Y., Feng, C., Shen, Y., Tian, D.: Foldingnet: Point cloud auto-encoder via deep grid deformation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 206–215 (2018)
77. Yi, L., Guibas, L., Hertzmann, A., Kim, V.G., Su, H., Yumer, E.: Learning hierarchical shape segmentation and labeling from online repositories. arXiv preprint arXiv:1705.01661 (2017)
78. Yi, L., Kim, V.G., Ceylan, D., Shen, I.C., Yan, M., Su, H., Lu, C., Huang, Q., Sheffer, A., Guibas, L.: A scalable active framework for region annotation in 3d shape collections. ACM Transactions on Graphics (TOG) **35**(6), 1–12 (2016)
79. Yi, L., Su, H., Guo, X., Guibas, L.J.: Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2282–2290 (2017)
80. Yin, G., Liu, B., Sheng, L., Yu, N., Wang, X., Shao, J.: Semantics disentangling for text-to-image generation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2327–2336 (2019)
81. Yu, F., Liu, K., Zhang, Y., Zhu, C., Xu, K.: Partnet: A recursive part decomposition network for fine-grained and hierarchical shape segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9491–9500 (2019)
82. Yumer, M.E., Chaudhuri, S., Hodgins, J.K., Kara, L.B.: Semantic shape editing using deformation handles. ACM Transactions on Graphics (TOG) **34**(4), 1–12 (2015)

83. Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., Metaxas, D.N.: Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In: Proceedings of the IEEE international conference on computer vision. pp. 5907–5915 (2017)
84. Zhao, B., Meng, L., Yin, W., Sigal, L.: Image generation from layout. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8584–8593 (2019)
85. Zou, C., Yumer, E., Yang, J., Ceylan, D., Hoiem, D.: 3d-prnn: Generating shape primitives with recurrent neural networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 900–909 (2017)

Appendix

A. More Details on the User Study

We show our user study interface in Figure 9. We ask the users to rank three algorithms from three aspects: part structure, geometry, overall.

B. More Qualitative Results

We present more qualitative results in Figure 10. Given the symbolic part trees as conditions, we show multiple diverse point clouds generated by our method.

C. 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 7 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.

D. Failure Cases and Future Works

Figure 8 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.

E. Part Tree Visualization for Figure 4

Figure 11 shows the eight part tree conditional inputs used for generating the point cloud shapes in Figure 4.

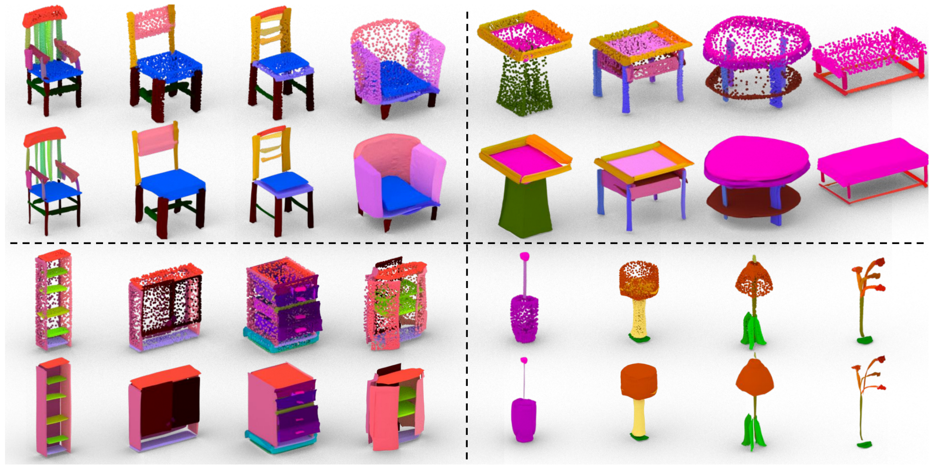


Fig. 7. Mesh Generation Results. The top rows show the generated shape point clouds and the bottom rows show the corresponding generated mesh results.

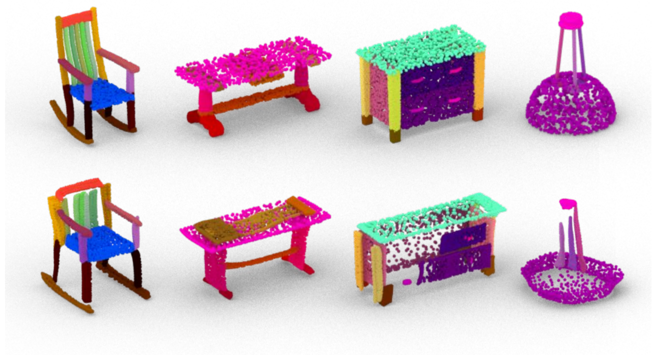
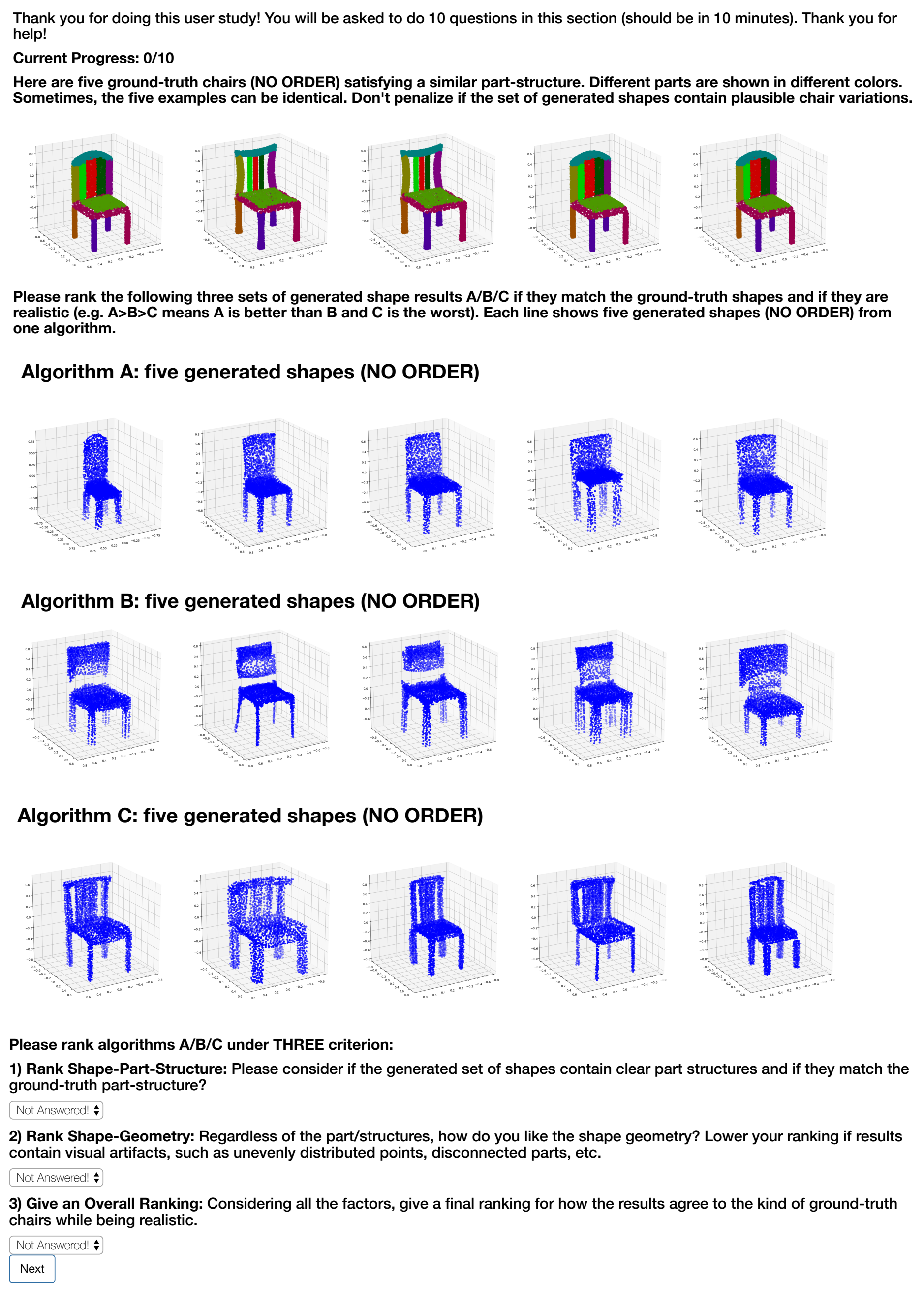


Fig. 8. failure cases. The top row shows the real shapes and the bottom row presents our generated point clouds.



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!

Fig. 9. User Study Interface.

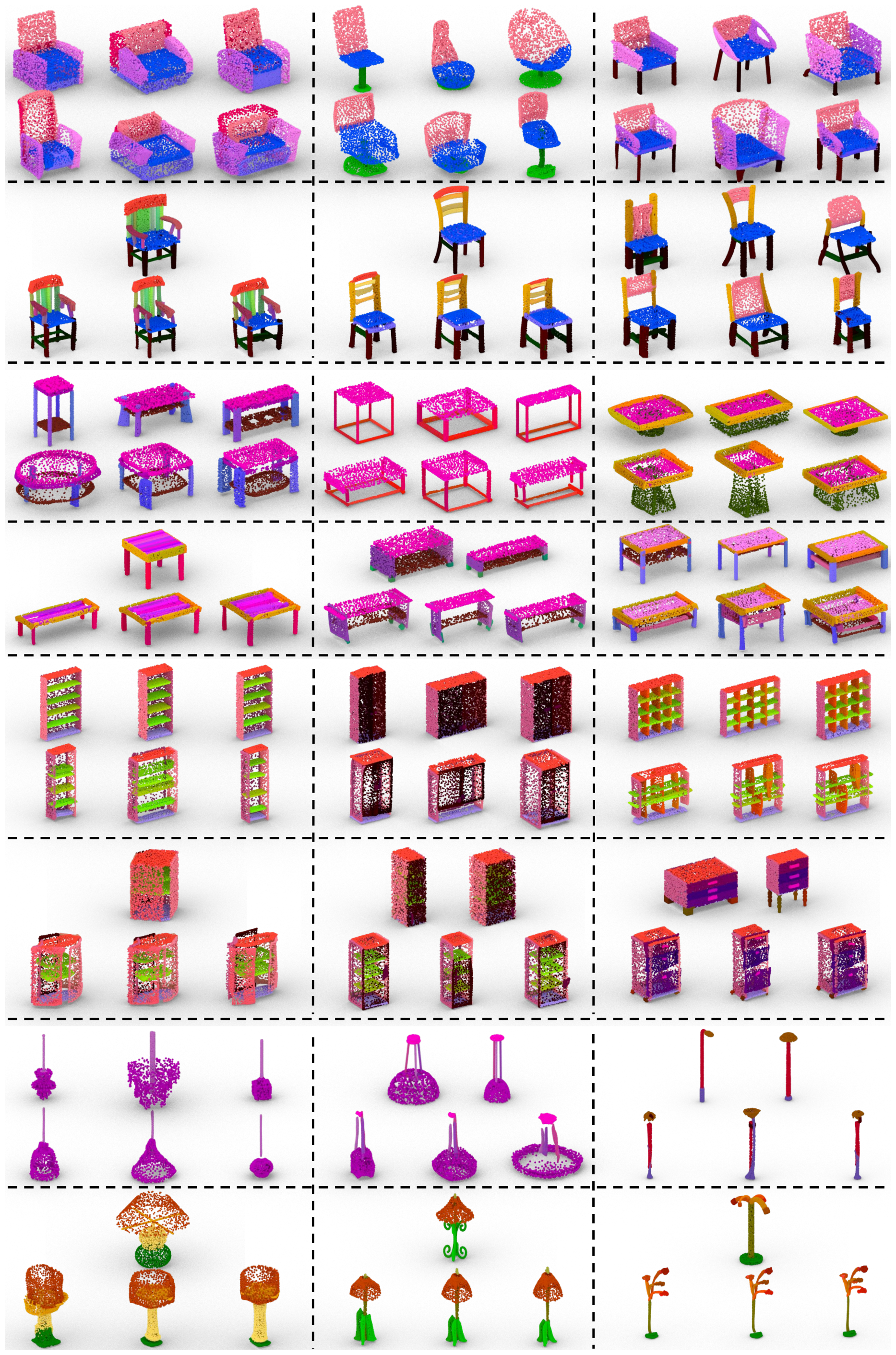


Fig. 10. Additional qualitative results. We show six more results for each of the four categories. For each block, the top row shows the real shapes and the bottom row shows our generated results.

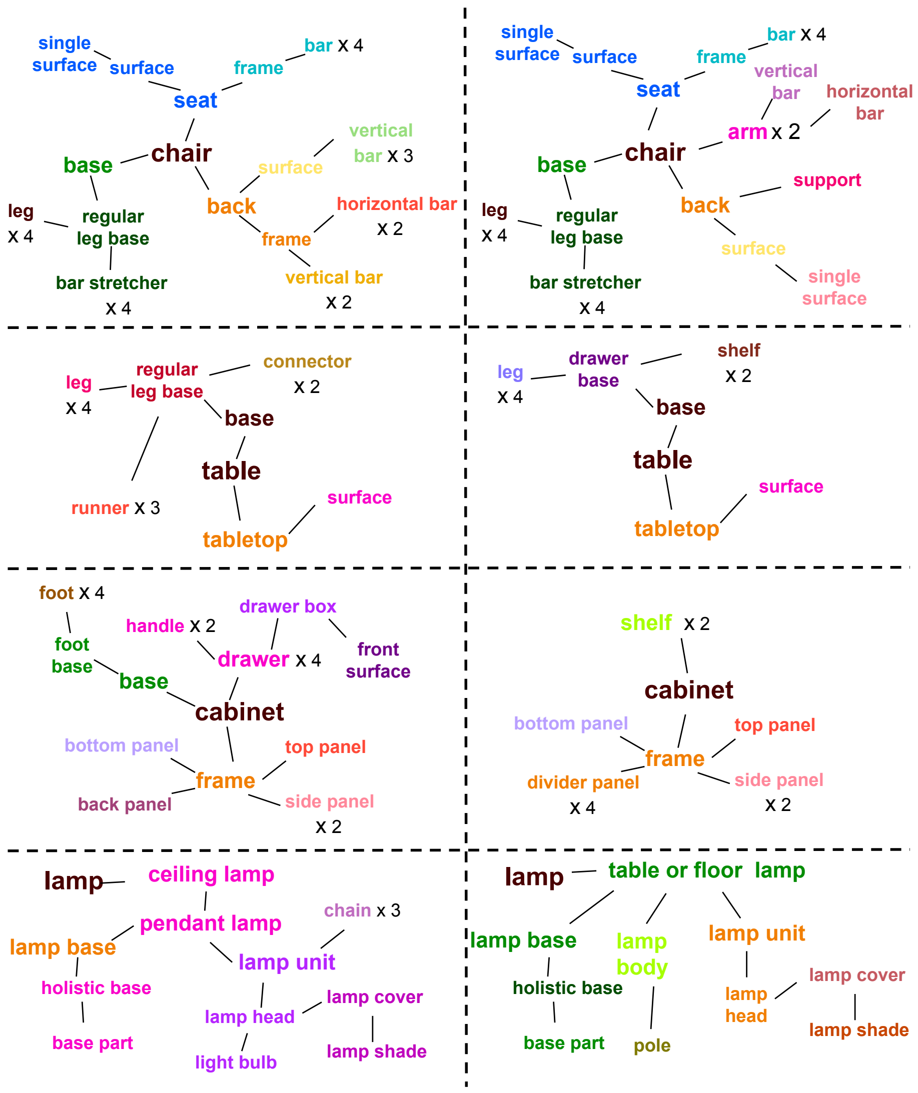


Fig. 11. Visualization for the Part Tree Conditions for Figure 4. Here we show the eight part tree conditional inputs used for generating the point cloud shapes in Figure 4.