PASTA: Controllable Part-Aware Shape Generation with Autoregressive Transformers

Songlin Li, Despoina Paschalidou, and Leonidas Guibas
Stanford University
353 Jane Stanford Way, Stanford, CA, 94305, USA
{svli97, paschald, guibas}@stanford.edu

Abstract

The increased demand for tools that automate the 3D content creation process led to tremendous progress in deep generative models that can generate diverse 3D objects of high fidelity. In this paper, we present PASTA, an autoregressive transformer architecture for generating high quality 3D shapes. PASTA comprises two main components: An autoregressive transformer that generates objects as a sequence of cuboidal primitives and a blending network, implemented with a transformer decoder that composes the sequences of cuboids and synthesizes high quality meshes for each object. Our model is trained in two stages: First we train our autoregressive generative model using only annotated cuboidal parts as supervision and next, we train our blending network using explicit 3D supervision, in the form of watertight meshes. Evaluations on various ShapeNet objects showcase the ability of our model to perform shape generation from diverse inputs e.g. from scratch, from a partial object, from text and images, as well size-guided generation, by explicitly conditioning on a bounding box that defines the object's boundaries. Moreover, as our model considers the underlying part-based structure of a 3D object, we are able to select a specific part and produce shapes with meaningful variations of this part. As evidenced by our experiments, our model generates 3D shapes that are both more realistic and diverse than existing part-based and non part-based methods, while at the same time is simpler to implement and train.

1. Introduction

The ability to generate realistic and diverse 3D shapes has the potential to significantly accommodate the workflow of artists and content creators and potentially enable new levels of creativity through "generative art" [5]. The tremendous progress in generative modelling and implicit-based representations gave rise to several works [10, 11, 28, 34, 92] that generate objects with high realism in terms of geometric details and texture. Nevertheless, as these pipelines represent objects holistically, i.e. without taking into consideration the underlying part-based structure of each object, they only support few interactive applications that typically require deep technical knowledge of each model. However, shape editing and manipulation involves controlling what parts of the object need to be changed. To enable this level of control, an active area of research proposes to consider the decomposition of shapes into parts [19, 27, 35, 37, 51, 65, 67, 110].

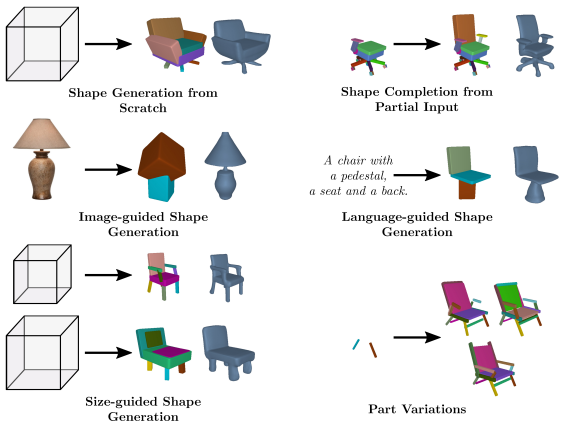

Figure 1. **Controllable Part-Aware 3D Shape Generation**. We propose a novel autoregressive architecture that can be used to perform several editing tasks, such as generating novel shapes from scratch, conditioned on a bounding box defining the object's boundaries, completing a 3D shape from a partial input, a text, an image or bounding boxes of different sizes, as well as generating plausible variations for specific parts of the object.

Existing part-based generative models, represent 3D shapes as a collection of simple shapes parametrized with cuboids [65, 67, 110, 122], spheres [35], implicit fields [37, 110] or more general handles [27, 55], and seek to synthesize new shapes in accordance to their underlying structural understanding of the object. For example, among the first to explore structure-based generative models were [50, 65] that utilized an autoencoder for generating structured shapes. Despite their impressive capabilities

on shape generation and interpolation neither can perform shape completion from a partial input or generate plausible part variations. Similarly, while [110, 122] can generate 3D shapes as a sequences of parts, they need to train a separate model for different editing tasks, namely the same model cannot perform both shape generation and shape completion, which makes their approach impractical.

To address these limitations, we devise PASTA, a novel part-aware generative model for 3D shapes. PASTA comprises two main components: An autoregressive transformer encoder that generates shapes as unordered sets of parts and a blending network, implemented as a transformer decoder that combines the part sequences and produces high-quality meshes. Each component of our architecture is trained independently. In particular, we optimize our autoregressive transformer to maximize the log-likelihood of all part arrangements in the dataset. Our supervision comes in the form of part labels and 3D cuboids that specify the per-part size and pose. Unlike existing autoregressive pipelines [79, 100] that are trained using teacher forcing, we train PASTA using scheduled sampling [6, 62] and showcase that it significantly improves the generation performance of our model. To train our blending network, we consider 3D supervision in the form of watertight meshes and optimize it to reconstruct 3D shapes as implicit occupancy fields [60]. We evaluate the performance of our model on several PartNet [66] objects and demonstrate that our model can produce more realistic and diverse 3D objects in comparison to both part-based [79, 110] and non part-based methods [14]. Furthermore, we showcase that our model can generate meaningful part arrangements conditioned on versatile user input Fig. 1 including but not limited to text and images.

In summary we make the following **contributions**: We propose the first part-aware generative model using an autoregressive transformer architecture. Our experiments on various PartNet objects [66] demonstrate that our model generates more diverse and plausible 3D shapes in comparison to part-based [79, 110] and non part-based methods [14]. Furthermore, our simple, yet effective architecture allows training a single model capable of performing several editing operations, such as generating new objects from scratch, generating part variations or completing partial shapes.

2. Related Work

3D Representations Learning-based approaches for 3D reconstruction employ a neural network that learns a function from the input to a mesh [33, 44, 52, 74, 106, 113], a point-cloud [2, 25, 43, 84, 98, 113], a voxel grid [7, 17, 26, 86, 87, 96, 111] or an implicit surface [14, 60, 61, 75, 89, 112]. Unlike explicit representations that discretize the output space, using voxels, points or mesh vertices, implicit representations represent shapes in the weights of a neural network

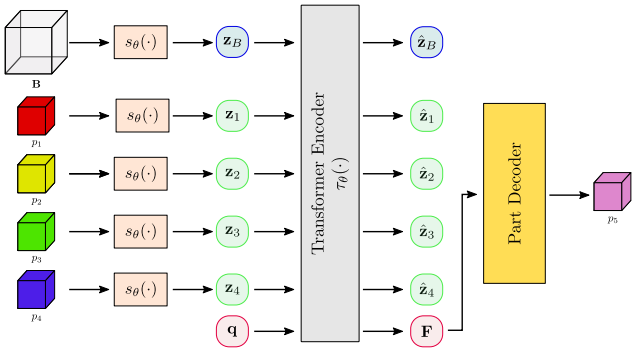

Figure 2. **Object Generator.** Given a sequence of N parts and a bounding box $\mathbf{B}$ defining the object boundaries, the *part encoder* $s_\theta(\cdot)$ maps each part p_j and the bounding box to an embedding vector. The bounding box's embedding vector $\mathbf{z}_B$, the per-part embeddings $\{\mathbf{z}_j\}_{i=1}^N$ and a learnable embedding vector $\mathbf{q}$ are passed to the *transformer decoder* that predicts a feature vector $\mathbf{F}$ used to predict the attributes of the next part in the sequence. The *part decoder* takes $\mathbf{F}$ and autoregressively predicts the attribute distributions that are used to sample the attributes for the next part.

that learns a mapping between a query point and a context vector to a signed distance value [4, 32, 61, 75, 97] or a binary occupancy value [14, 60]. As these methods require 3D supervision, several works propose combining them with surface [71, 116] or volumetric [63] rendering to learn the 3D object geometry and texture directly from images. In this work, we introduce a part-aware generative model that parametrizes shapes as an occupancy field [60].

Primitive-based Representations Shape abstraction techniques represent shapes using semantically consistent part arrangements and seek to recover the 3D geometry using simple shapes such as cuboids [22, 50, 65, 72, 101, 122], superquadrics [77, 78], convex solids [15, 18, 27], spheres [35] and 3D Gaussians [29]. In recent work, [80] proposed to represent objects as a family of homeomorphic mappings, parametrized with an Invertible Neural Network (INN) [21]. Likewise, [30] suggested to represent 3D objects using a structured set of implicit functions [29]. Another line of research employs primitives to recover the 3D geometry using CSG trees [93] or shape programs [23, 56, 99]. Very recently [114, 115], explored learning primitive-based representations only from images. Unlike these works that focus on recovering the object geometry as a collection of parts, we introduce a part-aware generative model that synthesizes novel objects as a set of cuboidal primitives.

3D Generative Models Generative Adversarial Networks (GANS) [31] have demonstrated impressive capabilities on several image synthesis [8, 16, 40, 45, 46] and editing [3, 16, 42, 54, 95, 105, 107, 121] tasks. However, adapting them to 3D content creation is non-trivial as they ignore the 3D nature of the world and hence, lack an understanding of the object's underlying geometry. To address this, 3D-aware GANs proposed to incorporate 3D representations such as voxel grids [36, 69, 70] in generative settings

or combine them with differentiable renderers [53, 118]. As an alternative, several works explored generating 3D shapes as octrees [41], pointclouds [2, 9, 51, 59, 113, 119], meshes [58, 68, 82, 83] and implicit functions [15, 60]. While these methods yield realistic geometries, they do not consider the part-based object structure. In contrast, we propose a part-aware generative model that generates shapes as an unordered set of cuboids, which are then combined and synthesize a high quality implicit shape.

Part-based Generative Models Our work falls into the category of part-based generative models. Zou et al. [122] was among the first to introduce a generative recurrent model, parametrized with LSTMs [39] in combination with a Mixture Density Network (MDN) to synthesize novel objects as a set of cuboids. Concurrently, Li et al. [50] proposed to represent shapes using a symmetry hierarchy, which defines how parts are recursively grouped by symmetry and assembled by connectivity [109]. In particular, they utilize an RNN and generate objects as bounding box layouts, which are then filled with voxelized parts. Likewise, StructureNet [65] utilizes a VAE [48] and generates novel shapes as n-ary graphs, where every node in the graph is associated with a bounding box. Note that while [65] can generate plausible new shapes, perform shape and part interpolations, their model cannot be utilized for completion from unconstrained inputs, e.g. a partial object. Furthermore, to perform shape editing, their model requires additional optimization steps in order to find a new shape in the latent space that satisfies a specific edit. Concurrently, [67], explored learning a latent space of structured shape differences, using pairs of structured shapes demonstrating a specific edit. Closely related to our work is PQ-NET [110] that generates shapes autoregressively using an RNN autoencoder. The RNN encoder takes a 3D shape segmented into parts and extracts per-part features, which are then fed to the decoder that sequentially predicts parts that reconstruct the input shape. To be able to generate new shapes, they train a latent GAN [2] on the latent space of the autoencoder. Moreover, to perform different editing tasks, they need to train different variants of their model. Instead, our formulation allows applying a single model trained for object completion on a variety of tasks. Furthermore, instead of using an RNN, we employ an autoregressive transformer that synthesizes objects as a sequence of cuboids, which are passed to our blending network to produce the final high-quality shape.

Autoregressive Transformers for Content Creation Transformer-based architectures [104] have been extensively utilized for various autoregressive tasks ranging from machine translation [73, 94] to image [13, 24, 47, 76, 100] and music [20] generation, as well as shape completion [64, 117] and indoor scene synthesis [79, 108]. Similar to ATISS [79], that is an autoregressive transformer for scene synthesis, we pose object synthesis as an autoregressive prediction problem and generate objects as a sequence of cuboids. However, unlike ATISS that is trained with teacher forcing, we train our model using scheduled sampling [6, 62] and showcase that it improves the generation capabilities of our model.

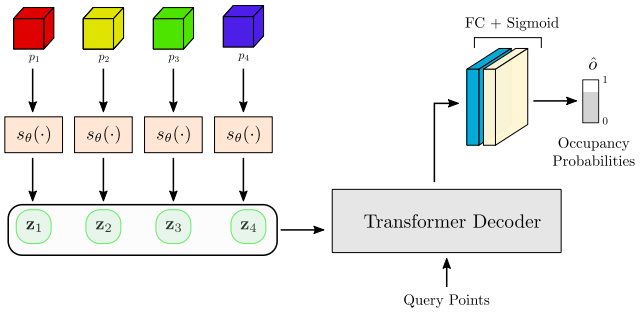

Figure 3. **Blending Network.** Given a sequence of N parts, the *part encoder* maps them into embedding vectors $\{\mathbf{z}_j\}_{i=1}^{N}$. We pass the per-part embedding vectors and a set of 3D query points $\mathcal{X}$ to the *transformer decoder* that predicts the occupancy probabilities for the query points.

3. Method

PASTA is a part-aware generative model for synthesizing 3D shapes. Our model comprises two main components that are trained independently: An *object generator* that sequentially generates objects as unordered sequences of labelled parts, where each part is parametrized using a 3D cuboidal primitive (Sec. 3.1), and a *blending network* that composes part sequences in a meaningful way and synthesizes high quality implicit shapes. The *object generator* is an autoregressive model trained to maximize the log-likelihood of all possible part arrangements in the dataset. We use part-level supervision in the form of part labels and 3D cuboids that define the size and the pose of each part (Sec. 3.2). The *blending network* is an occupancy network [60], implemented with a transformer decoder that takes a sequence of cuboids and a query 3D point and predicts whether this point is inside or outside the surface boundaries (Sec. 3.3). To train the blending network, we assume explicit 3D supervision in the form of a watertight mesh.

3.1. Object Parametrization

We define each object $\mathcal{S} = \{\mathcal{P}, \mathbf{B}\}$ using a collection of N parts, $\mathcal{P} = \{p_1, \ldots, p_N\}$, and a 3D bounding box $\mathbf{B}$ that specifies the object's boundaries. Each part is represented with a 3D labelled cuboid and is parametrized using four values describing its label, size, translation and rotation, $p_j = \{\mathbf{c}_j, \mathbf{s}_j, \mathbf{t}_j, \mathbf{o}_j\}$. To model the label of each part, i.e. the back or the arm of a chair, we use a categorical distribution defined over the total number of part labels in the dataset, whereas for the rest of the attributes we use a mixture of logistic distributions [90, 102]. In our setup, the rotation $\mathbf{o}_j \in \mathbb{R}^6$ is the 6D representation [120] of the

3D cuboid that contains the part. Likewise, the translation $\mathbf{t}_j \in \mathbb{R}^3$ is the center of the 3D cuboid containing the part and the size $\mathbf{s}_j \in \mathbb{R}^3$ is its width, height and depth. Unlike parts, the bounding box $\mathbf{B}$ is parameterized by 12 values without a label: three for the size along each axis, three for the translation (center of the box), and six for the rotation (using a 6D representation).

Similar to ATISS [79], we predict the components of p_j autoregressively, namely part label first, followed by translation, rotation and size. Hence, the probability of generating the j-th part, conditioned on the previous parts and box $\mathbf{B}$ becomes

$$\begin{aligned} p_\theta(p_j \mid p_{<j}, \mathbf{B}) = & p^c_\theta(\mathbf{c}_j | p_{<j}, \mathbf{B}) p^t_\theta(\mathbf{t}_j | \mathbf{c}_j, p_{<j}, \mathbf{B}) \\ & p^o_\theta(\mathbf{o}_j | \mathbf{c}_j, \mathbf{t}_j, p_{<j}, \mathbf{B}) p^s_\theta(\mathbf{s}_j | \mathbf{c}_j, \mathbf{t}_j, \mathbf{o}_j, p_{<j}, \mathbf{B}), \end{aligned} \tag{1}$$

where p^c_θ, p^t_θ, p^o_θ and p^s_θ are the probabilities for the respective attributes. Assuming a fixed ordering wrt. the part attributes is reasonable, as we want our model to consider the part label before predicting its pose and size. To compute the likelihood of generating an object $\mathcal{S}$, we estimate the likelihood of autoregressively generating its parts $\mathcal{P}$ using any order, as [79] demonstrated that not having a fixed ordering can be beneficial. Hence, the likelihood of generating an object $\mathcal{S}$ conditioned on a bounding box $\mathbf{B}$ is

$$p_\theta(\mathcal{S}|\mathbf{B}) = \sum_{\hat{\mathcal{P}} \in \pi(\mathcal{P})} \prod_{j \in \hat{\mathcal{P}}} p_\theta(p_j \mid p_{<j}, \mathbf{B}), \tag{2}$$

where $\pi(\cdot)$ is a permutation function that computes the set of permutations of all object parts and $\hat{\mathcal{P}}$ denotes an ordered part sequence.

3.2. Object Generator

The input to our *object generator* is a set of objects in the form of 3D labelled cuboids and their corresponding bounding boxes. We implement our generator using an autoregressive transformer architecture similar to ATISS. The transformer model takes as input a sequence of embedding vectors that represent the conditioning sequence and generates the features $\mathbf{F}$ that will be used to predict the attributes of the next part. We map the per-part attributes to embedding vectors using a *part encoder* and the features $\mathbf{F}$ to part attributes using a *part decoder*. Our model is illustrated in Fig. 2.

The part encoder network $s_\theta(\cdot)$ takes the attributes for each part $p_j = \{\mathbf{c}_j, \mathbf{s}_j, \mathbf{t}_j, \mathbf{o}_j\}$ and maps them to an embedding vector $\mathbf{z}_j$

$$\mathbf{z}_j = s_\theta\big([\lambda(\mathbf{c}_j); \gamma(\mathbf{s}_j); \gamma(\mathbf{t}_j); \gamma(\mathbf{o}_j)] \big), \tag{3}$$

where $\lambda(\cdot)$ is a learnable embedding, $\gamma(\cdot)$ is a positional encoding layer [104] that is applied separately on each attribute's dimension and $[\cdot\,;\cdot]$ denotes concatenation. To predict an embedding vector $\mathbf{z}_B$ for $\mathbf{B}$, we pass its attributes to $s_\theta(\cdot)$.

Similar to ATISS [79], we implement our transformer encoder $\tau_\theta(\cdot)$ as a multi-head attention transformer without positional encoding [104], as we want to model objects as unordered sets of parts. Our transformer encoder takes as input $\{\mathbf{z}_j\}_{j=1}^N$ the N embeddings for all parts in the sequence, $\mathbf{z}_B$ the embedding for the bounding box and a learnable embedding vector $\mathbf{q}$, which is used to predict the feature vector $\mathbf{F}$ that will be used to generate the next part in the sequence. More formally,

$$\mathbf{F} = \tau_\theta\left(\mathbf{z}_B, \{\mathbf{z}_j\}_{j=1}^N, \mathbf{q}\right). \tag{4}$$

The last component of the object generator is the part decoder that takes as input the feature vector $\mathbf{F}$ and autoregressively predicts the attributes of the next part to be generated. For the part label, we define a function $c_\theta(\cdot)$, implemented using a linear projection layer, that takes $\mathbf{F}$ and predicts the per-part label probability. We predict the size, translation and rotation in two-stages. First, we cluster the values of each attribute from the training set into 20 clusters using K-Means. Subsequently, we predict a cluster for each attribute, which is then used to predict the specific values. More formally, for the translation, we learn $t^{\text{coarse}}_\theta(\cdot)$ that predicts the per-cluster probability from $\mathbf{F}$ using a linear projection layer and $t^{\text{fine}}_\theta(\cdot)$ that predicts the $7 \times K$ parameters that define the mixture of logistics distribution for the translation. Specifically, we have K parameters for the mixing coefficients and $6 \times K$ for the means and variances. In a similar manner, we define $o^{\text{coarse}}_\theta(\cdot)$ and $o^{\text{fine}}_\theta(\cdot)$ to predict the $13 \times K$ parameters that define the mixture of logistics distribution for the rotation and $s^{\text{coarse}}_\theta(\cdot)$ and $s^{\text{fine}}_\theta(\cdot)$ that predict the $7 \times K$ parameters that define the mixture of logistic distribution for the size. To predict the part attributes in an autoregressive manner, we condition the prediction of each attribute to the values of the previously predicted ones. In practice, $t^{\text{coarse}}_\theta(\cdot)$, $t^{\text{fine}}_\theta(\cdot)$, $o^{\text{coarse}}_\theta(\cdot)$, $o^{\text{fine}}_\theta(\cdot)$, $s^{\text{coarse}}_\theta(\cdot)$ and $s^{\text{fine}}_\theta(\cdot)$ take as input $\mathbf{F}$ concatenated with the previously predicted attributes embedded by $\lambda(\cdot)$ and $\gamma(\cdot)$ from Eq. (3).

3.3. Blending Network

The input to the *blending network* is a sequence of labelled cuboids and a set of 3D query points $\mathcal{X}$, for which we want to predict their occupancy probabilities, namely whether they lie inside or outside the surface boundaries. In detail, our blending network consists of two main components: (i) a *part-encoder* that maps the part attributes into an embedding vector, which is implemented as discussed in Sec. 3.2 and (ii) a transformer decoder without self-attention that takes the part embeddings and the query points and predicts whether they are inside or outside the surface boundary (see Fig. 3). The transformer comprises only cross attention layers and MLPs, thus, each query 3D point attends to the per-part embeddings using cross attention, without attending to the other points. The transformer output is passed to a linear layer followed by a sigmoid non-linearity to get an occupancy probability for each query point. We follow common

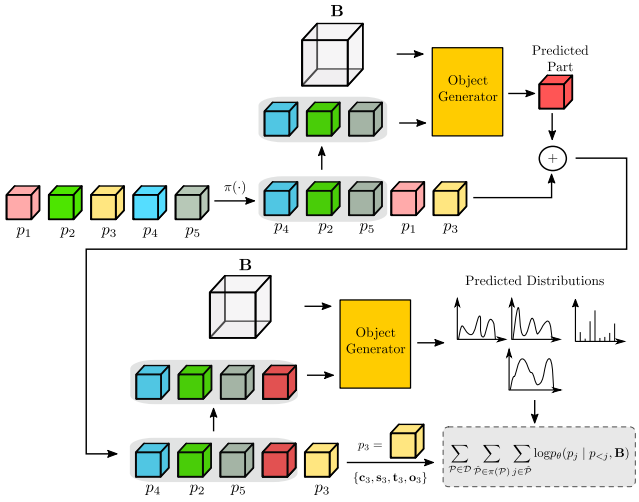

Figure 4. **Scheduled Sampling.** Given an object with N parts, we first randomly permute them and keep the first M parts (here $M = 3$). We pass them to the object generator that predicts the next part to be generated (red cube). The newly generated cuboid is appended to the initial sequence with the M objects and passed once again to the object generator, that predicts the next part to be generated. Our loss function from Eq. (5) is computed between the new part and the $M + 2$ part in the permutated sequence (yellow cube).

practice and before passing the query points to the decoder we map them to a higher dimensional space using positional encoding [104].

3.4. Training and Inference

Unlike prior autoregressive models [79, 88] that are trained with teacher forced embeddings, we train PASTA using scheduled sampling [6]. The key idea is that during training we feed our model with a mix of the teacher forced embeddings and the actual model's predictions from the previous generation step. In particular, we choose an object from the dataset and apply the permutation function $\pi(\cdot)$ on its elements. Next, we randomly select the first M objects and pass them to the object generator that is used to predict the next part. The newly generated part is appended to the initial sequence of M parts and passed once again to the object generator to predict the attribute distribution of the next part. Our model is trained to maximize the likelihood of the $M + 2$ object in the permuted sequence. A pictorial representation of our scheduled sampling is provided in Fig. 4. We follow common practice and during training, we train with both scheduled sampling and teacher forcing.

To train the object generator we follow [79] and maximize the likelihood of generating all possible part sequences in the dataset $\mathcal{D}$ in all possible permutations, as follows

$$\mathcal{L}(\theta) = \sum_{\mathcal{P}\in\mathcal{D}} \sum_{\hat{\mathcal{P}}\in\pi(\mathcal{P})} \sum_{j\in\hat{\mathcal{P}}} \log p_\theta(p_j \mid p_{<j}, \mathbf{B}). \quad (5)$$

For the mixture of logistic distributions, we use the discretized mixture of logistics loss as defined in [102]. To train the blending network, we assume 3D supervision in the form of a watertight mesh, which we use to generate a set of occupancy pairs $\mathcal{X} = \{\{\mathbf{x}_i, o_i\}\}_{i=1}^{V}$, namely a set of 3D points $\mathbf{x}_i$ and their occupancy labels o_i, denoting whether $\mathbf{x}_i$ lies inside or outside the object. We train the blending network using a classification loss between the predicted and the target occupancies. Note that the blending network is trained using ground-truth part sequences.

During inference, we start from a bounding box $\mathbf{B}$ and autoregressively sample the attribute values from the predicted distributions for the next part to be generated. Once a new part is generated, it is used in the next generation step until the *end symbol* is predicted. To indicate the end of sequence, we augment the part labels with an additional category, which we refer to as *end symbol*. Once a complete sequence of parts is generated, we pass it to the blending network that combines them into a single implicit shape.

3.5. Conditional Generation

Here, we discuss how PASTA can be used to perform language- and image-guided generation. Instead of conditioning the generation only on the bounding box B, we now condition also on a textual description of the shape to be generated. In particular, we utilize the pre-trained CLIP [85] model to extract text embeddings and pass them to the transformer encoder as an additional input. Note that during training the pre-trained CLIP text encoder remains frozen, namely is not optimized with the rest of our network. Once we train PASTA with text embeddings from CLIP, we can use it, without any re-training, also for image-guided generation. This is possible because the CLIP model has a joint latent space for text and images. While, in our experiments, we only demonstrate language- and image-guided generations, our model can be extended to other types of conditioning such as depth maps or pointclouds etc. by utilizing an appropriate encoder that generates embeddings from the input.

4. Experimental Evaluation

In this section, we provide an extensive evaluation of our method comparing it to relevant baselines. Additional results and implementation details are provided in the supplementary.

Datasets We report results on three PartNet [66] categories: *Chair, Table* and *Lamp*, which contain 4489, 5705, and 1554 shapes respectively. For the *Chair* category, there are 47 different types of parts, while for the *Lamp* and the *Table* we have 32 and 43 respectively. We train our model and the part-based baselines using the part annotations and train/test splits from PartNet. Moreover, for our model, we use the object bounding boxes specified in PartNet.

Baselines In our evaluation, we include PQ-NET [110] that is a generative model that generates 3D shapes using an

Method	Representation	MMD-CD (↓)				COV-CD (%, ↑)			
		Chair	Table	Lamp	All	Chair	Table	Lamp	All
IM-NET	Implicit	3.49	2.65	4.07	4.74	55.76	54.71	82.66	38.25
SPAGHETTI	Implicit	4.22	3.47	4.27	4.71	50.24	49.63	87.11	43.2
AutoSDF	Implicit	3.79	2.85	3.52	3.88	56.83	55.61	85.78	40.77
PQ-Net	Implicit Parts	4.49	3.94	3.73	4.82	48.93	46.44	77.56	40.41
ATISS	Cuboids	5.03	4.27	4.14	6.09	48.27	39.18	**91.56**	35.03
Ours-Parts	Cuboids	3.71	3.23	4.07	3.82	57.32	**57.14**	81.87	**51.01**
Ours	Implicit	**3.21**	**2.53**	**2.93**	**3.24**	**57.73**	56.80	88.88	50.89

Table 1. **Shape Generation.** We perform category-specific training (3rd-5th and 7th-9th columns) and joint training on multiple object categories (6th and 10th columns) and report the MMD-CD (↓) and the COV-CD (↑) between generated and real shapes from the test set.

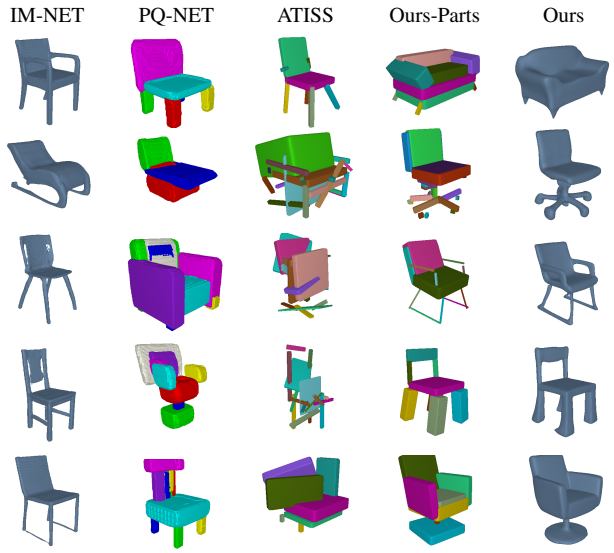

Figure 5. **Shape Generation Results on Chairs**. We show randomly generated chairs using our model, ATISS, PQ-NET and IM-NET.

RNN autoencoder, ATISS [79], which was originally introduced for scene synthesis but can be easily adapted to part-based object generation. In particular, instead of conditioning to a floor layout, we condition the generation on the object's bounding box, as for our model. Finally, we compare with IM-NET [14], which is an implicit-based generative model that does not reason about parts, hence not enabling part-level control. We conduct additional quantitative comparison against SPAGHETTI [37]: a self-supervised neural part-aware implicit field and AutoSDF: an autoregressive prior for various 3D shapes generation tasks [64]. Unlike PASTA, neither of these two methods directly utilizes semantic part sequences with variable lengths.

Metrics To evaluate the quality of the generated shapes, we report the Coverage Score (COV) and the Minimum Matching Distance (MMD) [2] using the Chamfer-L_2 distance (CD) between points sampled from real and generated shapes. To compute these metrics wrt. our part-based representation, we sample points on the surface of the union of the generated cuboids.

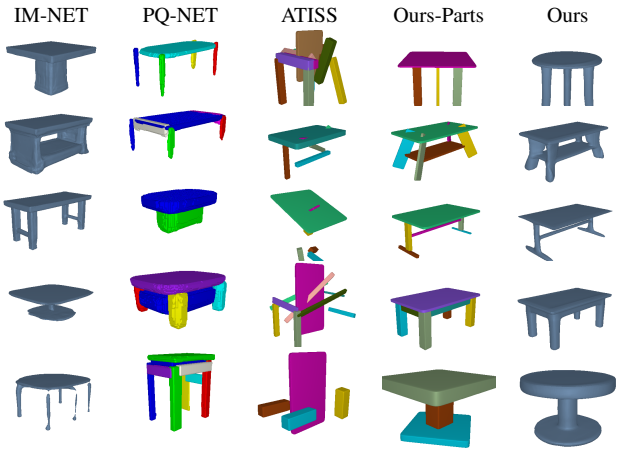

Figure 6. **Shape Generation Results on Tables**. We show randomly generated tables using our model, ATISS, PQ-NET and IM-NET.

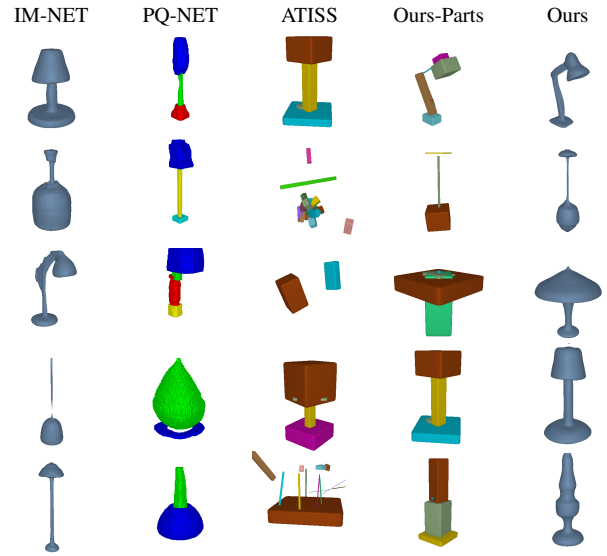

Figure 7. **Shape Generation Results on Lamps**. We showcase randomly generated lamps using our model, ATISS, PQ-NET and IM-NET.

4.1. Shape Generation

We evaluate the performance of our model on the shape generation task on chairs, tables and lamps and perform category-specific training for our model and the baselines. Conditioned on different bounding boxes, our model can successfully synthesize meaningful part arrangements (see 4th column in Fig. 5, Fig. 6 and Fig. 7), which are fed to our blending network that combines them and yields plausible 3D meshes (see 5th column in Fig. 5, Fig. 6 and Fig. 7). We observe that PASTA consistently generates diverse and realistic part arrangements (Ours-Parts) for all object categories, which are, in turn, converted into meshes (Ours) that faithfully capture the initial part representation with cuboids. On the contrary, ATISS struggles to synthesize meaningful part sequences, i.e. the synthesized part arrangements consist of parts positioned in unnatural posi-

Figure 8. **Shape Generation Results**. We show randomly generated chairs, tables and lamps using PASTA and our baselines, trained jointly on multiple objects categories.

Method	Representation	MMD-CD (↓)			COV-CD (%, ↑)		
		Chair	Table	Lamp	Chair	Table	Lamp
PQ-Net	Implicit Parts	4.69	3.64	**4.55**	35.77	42.31	53.33
ATISS	Cuboids	4.33	3.40	5.90	44.57	43.33	**60.88**
Ours-Parts	Cuboids	3.43	2.66	5.72	**50.49**	56.13	57.78
Ours	Implicit	**3.11**	**2.33**	5.58	49.00	**58.56**	51.00

Table 2. **Shape Completion.** We measure the MMD-CD (↓) and the COV-CD (↑) between the part-based representations of completed and real shapes from the test set.

tions, especially for the case of chairs and tables. While synthesized objects sampled from PQ-NET, IM-NET and other implicit baselines are more realistic than the part arrangements produced by ATISS, they lack diversity, as indicated by the coverage score in Tab. 1. Note that our model, even without the blending network (see Ours-Parts in Tab. 1), outperforms both PQ-NET and ATISS on chairs and tables on both metrics, while our complete architecture (see Ours in Tab. 1) outperforms also IM-NET and other implicit baselines on all object categories. For the case of lamps, we observe that our model outperforms all baselines wrt. MMD-CD, while performing on par with ATISS that achieves the highest score wrt. COV-CD. We hypothesize that ATISS performs better on the lamps category, as they typically consist of fewer components, hence making training and inference easier.

Next, we evaluate our model and the baselines' abilities to generate plausible 3D shapes when jointly trained on multiple object categories, without any class conditioning. Again our model consistently produces plausible part arrangements that are more realistic and diverse than our baselines, as validated by our quantitative analysis in Tab. 1

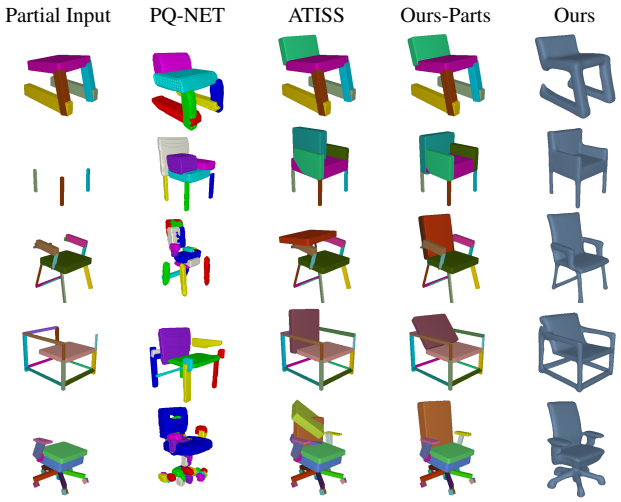

Figure 9. **Shape Completion Results on Chairs**. Starting from partial chairs, we show completions of our model, ATISS and PQ-NET.

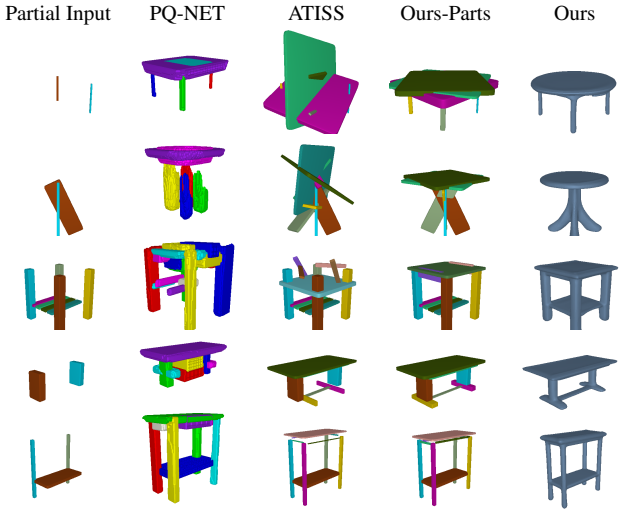

Figure 10. **Shape Completion Results on Tables**. Starting from partial tables, we show completions of our model, ATISS and PQ-NET.

(see 6th and 10th column). Fig. 8 provides a qualitative comparison of six objects generated with our model and the baselines.

4.2. Shape Completion

Starting from an incomplete sequence of parts, we evaluate whether our model and our baselines can complete the input sequence in a meaningful way. For this experiment, we only consider our part-based baselines and all models are trained in a category-specific manner. To ensure a fair comparison to PQ-NET, which is trained with 3D parts of arbitrary geometries, instead of conditioning their generation on the partial set of cuboids (illustrated in the 1st column of Fig. 9, Fig. 10 and Fig. 11), that is used for PASTA and ATISS, we utilize the corresponding 3D parts, that were used dur-

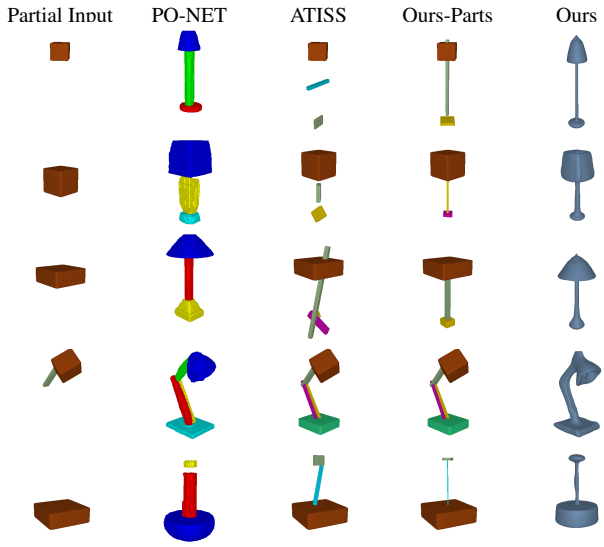

Figure 11. **Shape Completion Results on Lamps**. Starting from partial lamps, we show completions of our model, ATISS and PQ-NET.

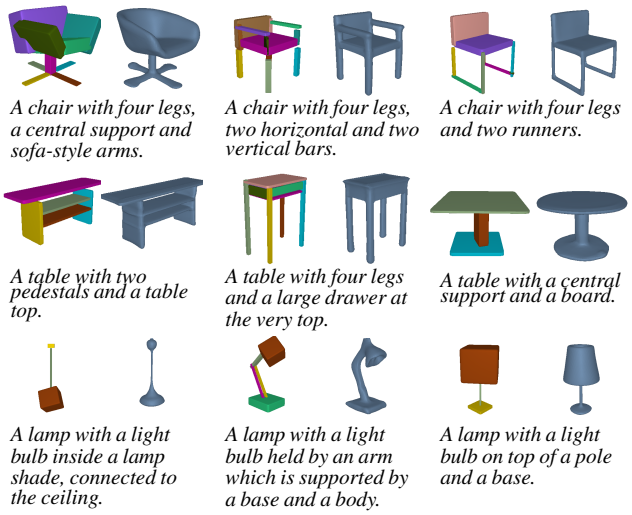

Figure 12. **Text-guided Shape Generation**. Given different text descriptions our model can generate plausible 3D shapes of chairs, tables, and lamps.

ing PQ-NET's training. From our qualitative evaluation, we observe that both ATISS and PQ-NET tend to generate non-realistic part arrangements, especially for the case of chairs (see unnatural back part for the 3rd and 6th chairs in the 3rd column of Fig. 9) and tables (see missing leg in the 2nd table in the 2nd column of Fig. 10). For the easier case of lamps (see Fig. 11), we observe that all methods can complete the partial object in a meaningful way. Note that, as PQ-NET relies on a sequence-to-sequence autoencoder to learn the part arrangements, there is no guarantee that the completed shape will contain the parts used for conditioning (see 3rd+4th row in Fig. 9, 2nd row in Fig. 10). Moreover, while for PASTA, the same model is used for both object completion and object generation, PQ-NET requires training a different model to perform the completion task.

We show that our generations are consistently valid and diverse. The quantitative results for this experiment are summarized in Tab. 2. We note that our model outperforms all baselines both on chairs and tables wrt. both metrics. For the case of lamps, ATISS outperforms all methods wrt. COV-CD, while being worse than all in terms of MMD-CD. To demonstrate that PASTA generates diverse part arrangements, we also visualize three generated completions of our model conditioned on the same partial input in Supplement 9.2. We observe that our generations are consistently valid and diverse.

4.3. Applications

In this section, we present several applications of our model, such as conditional generation from text and images. In both experiments, our model is trained in a category-specific manner.

Language-guided Generation We use the part labels provided in PartNet [66] and generate utterances that describe the part-based structure of each object. We train a variant of our model that conditions on CLIP [85] embeddings produced from our textual descriptions in addition to the object bounding box as described in Sec. 3.5. In Fig. 12, we provide text-guided generations of our model and observe that they consistently match the input text (e.g. the table with the two pedestals or the lamp connected to the ceiling).

Image-guided Generation: We showcase image-guided generations using the above language-guided trained model without any retraining in Supplement 9.5. We take advantage of the CLIP's joint latent space, and condition our model on image embeddings. Although PASTA was never trained with images, we show that it generates shapes of various object categories that faithfully match the input image. The recovered parts capture fine geometric details, such as the four horizontal connections of the first chair.

5. Conclusion

We introduced PASTA a part-aware generative model for 3D shapes. Our architecture consists of two main components: the *object generator* that autoregressively generates objects as sequences of labelled cuboids and the *blending network* that combines a sequence of cuboidal primitives and synthesizes an implicit shape. Unlike traditional autoregressive models that are trained with teacher forcing, we demonstrate that relying on scheduled sampling [6] improves the generation performance of our model. Our experiments, showcase that PASTA generates more meaningful part arrangements and plausible 3D objects than both part-based [79, 110] and non part-based generative models [14].

References

[1] Martín Abadi. Tensorflow: learning functions at scale. In *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*, 2016. 15

[2] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. Learning representations and generative models for 3d point clouds. 2018. 2, 3, 6, 14, 15

[3] Yazeed Alharbi and Peter Wonka. Disentangled image generation through structured noise injection. In *CVPR*, pages 5133–5141, 2020. 2

[4] Matan Atzmon and Yaron Lipman. SAL: sign agnostic learning of shapes from raw data. In *CVPR*, pages 2562–2571, 2020. 2

[5] Jason Bailey. The tools of generative art, from flash to neural networks. *Art in America*, 8, 2020. 1

[6] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *NeurIPS*, 2015. 2, 3, 5, 8

[7] André Brock, Theodore Lim, James M. Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. 1608.04236, 2016. 2

[8] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019. 2

[9] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge J. Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *ECCV*, 2020. 3

[10] Eric R. Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. Pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021. 1

[11] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J. Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3d generative adversarial networks. In *CVPR*, 2022. 1

[12] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. 1512.03012, 2015. 16

[13] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. 2020. 3

[14] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 2, 6, 8, 15, 16, 17, 19, 20, 21

[15] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. Bsp-net: Generating compact meshes via binary space partitioning. In *CVPR*, pages 42–51, 2020. 2, 3

[16] Yunjey Choi, Min-Je Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018. 2

[17] Christopher Bongsoo Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, 2016. 2

[18] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. Cvxnets: Learnable convex decomposition. *CVPR*, 2020. 2

[19] Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. Learning elementary structures for 3d shape generation and matching. In *NeurIPS*, 2019. 1

[20] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. 2020. 3

[21] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: non-linear independent components estimation. In *ICLR*, 2015. 2

[22] Anastasia Dubrovina, Fei Xia, Panos Achlioptas, Mira Shalah, Raphaël Groscot, and Leonidas J. Guibas. Composite shape modeling via latent space factorization. In *ICCV*, pages 8139–8148. IEEE, 2019. 2

[23] Kevin Ellis, Daniel Ritchie, Armando Solar-Lezama, and Joshua B. Tenenbaum. Learning to infer graphics programs from hand-drawn images. In *NeurIPS*, 2018. 2

[24] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. 2021. 3

[25] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. *CVPR*, 2017. 2

[26] Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. 2017. 2

[27] Matheus Gadelha, Giorgio Gori, Duygu Ceylan, Radomír Mech, Nathan Carr, Tamy Boubekeur, Rui Wang, and Subhransu Maji. Learning generative models of shape handles. In *CVPR*, 2020. 1, 2

[28] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. GET3D: A generative model of high quality 3d textured shapes learned from images. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1

[29] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *ICCV*, 2019. 2

[30] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas A. Funkhouser. Local deep implicit functions for 3d shape. In *CVPR*, 2020. 2

[31] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 2, 15

[32] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. 2020. 2

[33] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. AtlasNet: A papier-mâché approach to learning 3d surface generation. In *CVPR*, 2018. 2

[34] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d aware generator for high-resolution image synthesis. In *ICLR*, 2022. 1

[35] Zekun Hao, Hadar Averbuch-Elor, Noah Snavely, and Serge J. Belongie. Dualsdf: Semantic shape manipulation using a two-level representation. In *CVPR*, 2020. 1, 2, 24

[36] Philipp Henzler, Niloy J Mitra, , and Tobias Ritschel. Escaping plato's cave: 3d shape from adversarial rendering. In *ICCV*, 2019. 2

[37] Amir Hertz, Or Perel, Raja Giryes, Olga Sorkine-Hornung, and Daniel Cohen-Or. SPAGHETTI: editing implicit shapes through part aware generation. *ACM Trans. on Graphics*, 2022. 1, 6, 15, 24

[38] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. 18

[39] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. 9(8):1735–1780, 1997. 3

[40] Xinyu Huang, Peng Wang, Xinjing Cheng, Dingfu Zhou, Qichuan Geng, and Ruigang Yang. The ApolloScape Open Dataset for Autonomous Driving and its Application. 1803.06184, 2018. 2

[41] Moritz Ibing, Gregor Kobsik, and Leif Kobbelt. Octree transformer: Autoregressive 3d shape generation on hierarchically structured sequences. 2021. 3

[42] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 2

[43] Li Jiang, Shaoshuai Shi, Xiaojuan Qi, and Jiaya Jia. GAL: geometric adversarial loss for single-view 3d-object reconstruction. In *ECCV*, 2018. 2

[44] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018. 2

[45] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 2

[46] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. 2020. 2

[47] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. 2020. 3, 13

[48] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 3, 13, 14

[49] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 2014. 15

[50] Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao (Richard) Zhang, and Leonidas J. Guibas. GRASS: generative recursive autoencoders for shape structures. *ACM Trans. on Graphics*, 36(4), 2017. 1, 2, 3

[51] Ruihui Li, Xianzhi Li, Ka-Hei Hui, and Chi-Wing Fu. SP-GAN: sphere-guided 3d shape generation and manipulation. *ACM Trans. on Graphics*, 2021. 1, 3

[52] Yiyi Liao, Simon Donne, and Andreas Geiger. Deep marching cubes: Learning explicit surface representations. In *CVPR*, 2018. 2

[53] Yiyi Liao, Katja Schwarz, Lars M. Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. *CVPR*, 2020. 3

[54] Huan Ling, Karsten Kreis, Daiqing Li, Seung Wook Kim, Antonio Torralba, and Sanja Fidler. Editgan: High-precision semantic image editing. In *NeurIPS*, 2021. 2

[55] Minghua Liu, Minhyuk Sung, Radomír Mech, and Hao Su. Deepmetahandles: Learning deformation meta-handles of 3d meshes with biharmonic coordinates. In *CVPR*, 2021. 1

[56] Yunchao Liu, Zheng Wu, Daniel Ritchie, William T Freeman, Joshua B Tenenbaum, and Jiajun Wu. Learning to describe scenes with programs. In *ICLR*, 2019. 2

[57] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM Trans. on Graphics*, 1987. 16

[58] Andrew Luo, Tianqin Li, Wen-Hao Zhang, and Tai Sing Lee. Surfgen: Adversarial 3d shape synthesis with explicit surface discriminators. In *ICCV*, 2021. 3

[59] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *CVPR*, 2021. 3

[60] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 2, 3, 16

[61] Mateusz Michalkiewicz, Jhony K Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Implicit surface representations as layers in neural networks. In *ICCV*, 2019. 2

[62] Tsvetomila Mihaylova and André F. T. Martins. Scheduled sampling for transformers. 2019. 2, 3

[63] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2, 25

[64] Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. Autosdf: Shape priors for 3d completion, reconstruction and generation. In *CVPR*, 2022. 3, 6, 15

[65] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy Mitra, and Leonidas Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. In *ACM Trans. on Graphics*, 2019. 1, 2, 3, 16

[66] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *CVPR*, 2019. 2, 5, 8, 13, 15, 17, 18

[67] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy J. Mitra, and Leonidas J. Guibas. Structedit: Learning structural shape variations. In *CVPR*, 2020. 1, 3

[68] Charlie Nash, Yaroslav Ganin, S. M. Ali Eslami, and Peter W. Battaglia. Polygen: An autoregressive generative model of 3d meshes. 2020. 3

[69] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *ICCV*, 2019. 2

[70] Thu Nguyen-Phuoc, Christian Richardt, Long Mai, Yong-Liang Yang, and Niloy J. Mitra. Blockgan: Learning 3d

object-aware scene representations from unlabelled images. 2020. 2

[71] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *CVPR*, 2020. 2

[72] Chengjie Niu, Jun Li, and Kai Xu. Im2struct: Recovering 3d shape structure from a single RGB image. In *CVPR*, 2018. 2

[73] Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, 2018. 3

[74] Junyi Pan, Xiaoguang Han, Weikai Chen, Jiapeng Tang, and Kui Jia. Deep mesh reconstruction from single RGB images via topology modification networks. In *ICCV*, 2019. 2

[75] Jeong Joon Park, Peter Florence, Julian Straub, Richard A. Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2

[76] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. 2018. 3

[77] Despoina Paschalidou, Ali Osman Ulusoy, and Andreas Geiger. Superquadrics revisited: Learning 3d shape parsing beyond cuboids. In *CVPR*, 2019. 2, 14

[78] Despoina Paschalidou, Luc van Gool, and Andreas Geiger. Learning unsupervised hierarchical part decomposition of 3d objects from a single rgb image. In *CVPR*, 2020. 2

[79] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. Atiss: Autoregressive transformers for indoor scene synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 3, 4, 5, 6, 8, 13, 14, 15, 17, 19, 20, 21, 23, 24, 25

[80] Despoina Paschalidou, Angelos Katharopoulos, Andreas Geiger, and Sanja Fidler. Neural parts: Learning expressive 3d shape abstractions with invertible neural networks. In *CVPR*, 2021. 2

[81] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. 1606.02147, 2016. 15, 16

[82] Dario Pavllo, Graham Spinks, Thomas Hofmann, Marie-Francine Moens, and Aurélien Lucchi. Convolutional generation of textured 3d meshes. 2020. 3

[83] Dario Pavllo, Jonas Kohler, Thomas Hofmann, and Aurélien Lucchi. Learning generative models of textured 3d meshes from real-world images. In *ICCV*, 2021. 3

[84] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 2

[85] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. 5, 8, 16, 21, 22

[86] Danilo Jimenez Rezende, S. M. Ali Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. Unsupervised learning of 3d structure from images. In *NeurIPS*, 2016. 2

[87] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *CVPR*, 2017. 2

[88] Daniel Ritchie, Kai Wang, and Yu-An Lin. Fast and flexible indoor scene synthesis via deep convolutional generative models. In *CVPR*, 2019. 5

[89] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *ICCV*, 2019. 2

[90] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *ICLR*, 2017. 3

[91] Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.*, 1997. 15

[92] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. 2020. 1

[93] Gopal Sharma, Rishabh Goyal, Difan Liu, Evangelos Kalogerakis, and Subhransu Maji. Csgnet: Neural shape parser for constructive solid geometry. In *CVPR*, 2018. 2

[94] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, 2018. 3

[95] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *CVPR*, 2020. 2

[96] David Stutz and Andreas Geiger. Learning 3d shape completion from laser scan data with weak supervision. In *CVPR*, 2018. 2, 16

[97] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles T. Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *CVPR*, 2021. 2

[98] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019. 2

[99] Yonglong Tian, Andrew Luo, Xingyuan Sun, Kevin Ellis, William T Freeman, Joshua B Tenenbaum, and Jiajun Wu. Learning to infer and execute 3d shape programs. In *ICLR*, 2019. 2

[100] Shubham Tulsiani and Abhinav Gupta. Pixeltransformer: Sample conditioned signal generation. 2021. 2, 3

[101] Shubham Tulsiani, Hao Su, Leonidas J. Guibas, Alexei A. Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *CVPR*, 2017. 2

[102] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13-15 September 2016*, 2016. 3, 5

[103] Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 15

[104] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017. 3, 4, 5, 13, 18

[105] Binxu Wang and Carlos R. Ponce. A geometric analysis of deep generative image models and its applications. In *ICLR*, 2021. 2

[106] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *ECCV*, 2018. 2

[107] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018. 2

[108] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. Sceneformer: Indoor scene generation with transformers. 2020. 3

[109] Yanzhen Wang, Kai Xu, Jun Li, Hao Zhang, Ariel Shamir, Ligang Liu, Zhi-Quan Cheng, and Yueshan Xiong. Symmetry hierarchy of man-made objects. In *EUROGRAPHICS*, 2011. 3

[110] Rundi Wu, Yixin Zhuang, Kai Xu, Hao Zhang, and Baoquan Chen. PQ-NET: A generative part seq2seq network for 3d shapes. In *CVPR*, 2020. 1, 2, 3, 5, 8, 15, 16, 17, 19, 20, 21, 23, 24, 25

[111] Haozhe Xie, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, and Shengping Zhang. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. In *ICCV*, 2019. 2

[112] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomír Mech, and Ulrich Neumann. DISN: deep implicit surface network for high-quality single-view 3d reconstruction. In *NeurIPS*, 2019. 2

[113] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge J. Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *ICCV*, 2019. 2, 3

[114] Chun-Han Yao, Wei-Chih Hung, Varun Jampani, and Ming-Hsuan Yang. Discovering 3d parts from image collections. In *ICCV*, 2021. 2

[115] Chun-Han Yao, Wei-Chih Hung, Yuanzhen Li, Michael Rubinstein, Ming-Hsuan Yang, and Varun Jampani. LASSIE: learning articulated shapes from sparse image ensemble via 3d part discovery. abs/2207.03434, 2022. 2

[116] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Ronen Basri, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. In *NeurIPS*, 2020. 2

[117] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *ICCV*, 2021. 3

[118] Yuxuan Zhang, Wenzheng Chen, Huan Ling, Jun Gao, Yinan Zhang, Antonio Torralba, and Sanja Fidler. Image gans meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. In *ICLR*, 2021. 3

[119] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *ICCV*, 2021. 3

[120] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *CVPR*, 2019. 3

[121] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *ICCV*, 2017. 2

[122] Chuhang Zou, Ersin Yumer, Jimei Yang, Duygu Ceylan, and Derek Hoiem. 3d-prnn: Generating shape primitives with recurrent neural networks. In *ICCV*, 2017. 1, 2, 3