Generating Part-Aware Editable 3D Shapes without 3D Supervision

Konstantinos Tertikas[*1,3] Despoina Paschalidou[2] Boxiao Pan[2]
Jeong Joon Park[2] Mikaela Angelina Uy[2] Ioannis Emiris[3,1] Yannis Avrithis[4] Leonidas Guibas[2]
[1]National and Kapodistrian University of Athens [2]Stanford University
[3]Athena RC, Greece [4]Institute of Advanced Research in Artificial Intelligence (IARAI)

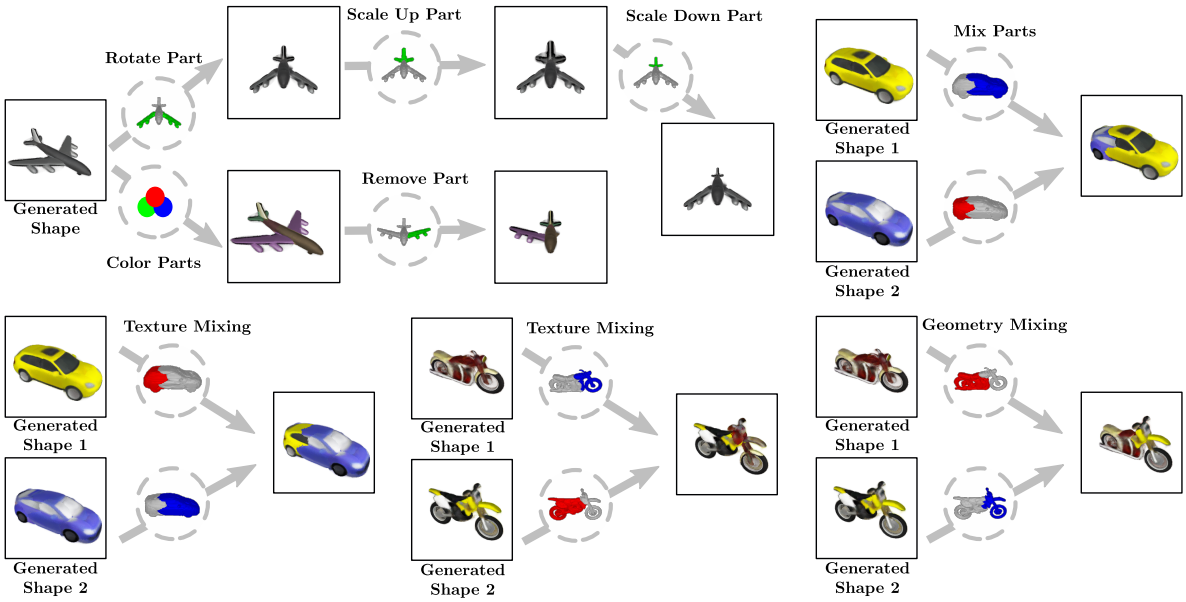

Figure 1. **Part-Aware Controllable 3D Shape Generation and Editing**. We address the task of part-aware 3D shape generation and editing without explicit 3D supervision. Prior part-aware generative models [34, 40] assume 3D supervision, at training, and only allow changing the shape of the object. In this work, we introduce PartNeRF, a generative model capable of editing the shape and appearance of generated shapes that are parametrized as a collection of locally defined NeRFs.

Abstract

Impressive progress in generative models and implicit representations gave rise to methods that can generate 3D shapes of high quality. However, being able to locally control and edit shapes is another essential property that can unlock several content creation applications. Local control can be achieved with part-aware models, but existing methods require 3D supervision and cannot produce textures. In this work, we devise PartNeRF, a novel part-aware generative model for editable 3D shape synthesis that does not require any explicit 3D supervision. Our model generates objects as a set of locally defined NeRFs, augmented with an affine transformation. This enables several editing operations such as applying transformations on parts, mixing parts from different objects etc. To ensure distinct, manipulable parts we enforce a hard assignment of rays to parts that makes sure that the color of each ray is only determined by a single NeRF. As a result, altering one part does not affect the appearance of the others. Evaluations on various ShapeNet categories demonstrate the ability of our model to generate editable 3D objects of improved fidelity, compared to previous part-based generative approaches that require 3D supervision or models relying on NeRFs.

1. Introduction

Generating realistic and editable 3D content is a long-standing problem in computer vision and graphics that has recently gained more attention due to the increased demand for 3D objects in AR/VR, robotics and gaming applications.

[*]Work done during internship at Stanford.

However, manual creation of 3D models is a laborious endeavor that requires technical skills from highly experienced artists and product designers. On the other hand, editing 3D shapes, typically involves re-purposing existing 3D models, by manually changing faces and vertices of a mesh and modifying its respective UV-map [95]. To accommodate this process, several recent works introduced generative models that go beyond generation and allow editing the generated instances [13,18,52,55,62,77,101,116,117,124]. Shape editing involves making *local changes* on the shape and the appearance of different parts of an object. Therefore, having a basic understanding of the decomposition of the object into parts facilitates controlling *what to edit*.

While Generative Adversarial Networks (GANs) [30] have emerged as a powerful tool for synthesizing photorealistic images [7, 15, 16, 47–49], scaling them to 3D data is non-trivial as they ignore the physics of image formation process.To address this, 3D-aware GANs incorporate 3D representations such as voxel grids [38, 72, 75] or combine them with differentiable renderers [57, 126]. While they faithfully recover the geometry and appearance, they do not allow changing specific parts of the object.

Inspired by the rapid evolution of neural implicit rendering techniques [68], recent works [8, 32, 77, 96, 114] proposed to combine them with GANs in order to allow for multi-view-consistent generations of high quality. Despite their impressive performance on novel view synthesis, their editing capabilities are limited. To this end, editing operations in the latent space have been explored [21, 42, 62, 115, 124] but these approaches lack intuitive control over the shape. By decomposing shapes into parts, other works facilitate structure-aware shape manipulations [40,70,88,111]. However, they require 3D supervision during training and can only operate on textureless shapes.

To address these limitations, we devise PartNeRF, a novel part-aware generative model, implemented as an auto-decoder [5]. Our model enables part-level control, which facilitates various editing operations on the shape and appearance of the generated instance. These operations include rigid and non-rigid transformations on the object parts, part mixing from different objects, removing/adding parts and editing the appearance of specific parts of the object.

Our key idea is to represent objects using a set of locally defined Neural Radiance Fields (NeRFs) that are arranged such that the object can be plausibly rendered from a novel view. To enable part-level control, we enforce a *hard assignment* between parts and rays that ensures that altering one part does not affect the shape and appearance of the others. Our model does not require 3D supervision; we only assume supervision from images and object masks captured from known cameras. We evaluate PartNeRF on various ShapeNet categories and demonstrate that it generates textured shapes of higher fidelity than both part-based as well as NeRF-based generative models. Furthermore, we showcase several editing operations, not previously possible.

In summary, we make the following **contributions**: We propose the first part-aware generative model that parametrizes parts as NeRFs and can generate editable 3D shapes. Unlike prior part-based approaches, our model does not require explicit 3D supervision and can generate textured shapes. Compared to NeRF-based generative models, our work is the first that reasons about parts and hence enables operations both on the shape and the texture of the generated object. Code and data is available at https://ktertikas.github.io/part_nerf.

2. Related Work

We now discuss the most relevant literature on 3D generative models in the context of generating editable shapes.

Neural Implicit Representations: Neural Implicit Representations [12, 66, 83] have demonstrated impressive capabilities on various tasks ranging from 3D reconstruction with [81, 82, 93, 94] and without texture [2, 3, 12, 14, 31, 43, 66,67,83,87,92,113] to video encoding [11], 3D-aware generative modelling [8, 9, 20, 32, 65, 77, 96], inverse graphics [78, 120] and novel view synthesis [39, 68, 91, 110, 123]. In contrast to explicit representations *i.e*. point clouds, meshes and voxels, implicit representations encode the shape's geometry and appearance in the weights of a neural network. Among the most extensively used implicit-based models are NeRFs [68], which combine a neural network with volumetric rendering [45] to perform novel view synthesis. Due to their compelling results, numerous works have been introduced to improve the training and rendering time [27, 59, 61, 71, 89, 90, 103, 122, 125], the underlying geometry [82, 109], to better handle lighting variations [4, 6, 64, 99] and to encode shape priors for better generalization [39, 105, 123] and generation [9, 96]. For a thorough overview on NeRF-based approaches we refer readers to [104, 112]. In our work, we introduce a generative model for editable 3D shapes with texture. Specifically, we parametrize objects as a structured set of local NeRFs that are trained from posed images and object masks.

3D-Aware Image Generation: Incorporating 3D representations in generative settings [20,23,32,35–38,63,65,73,74, 76, 127] has significantly improved the quality of the generated images and increased control over various aspects of the image formation process. Likewise, radiance fields have been combined with GANs to allow for photorealistic image synthesis of objects [9, 96] and scenes [77]. More recently, [8] introduced a triplane-based architecture that leverages both implicit and explicit representations and can generate high resolution images. Concurrently, [25] combined differentiable surface modelling [24] with a differentiable renderer to generate high-quality textured meshes.

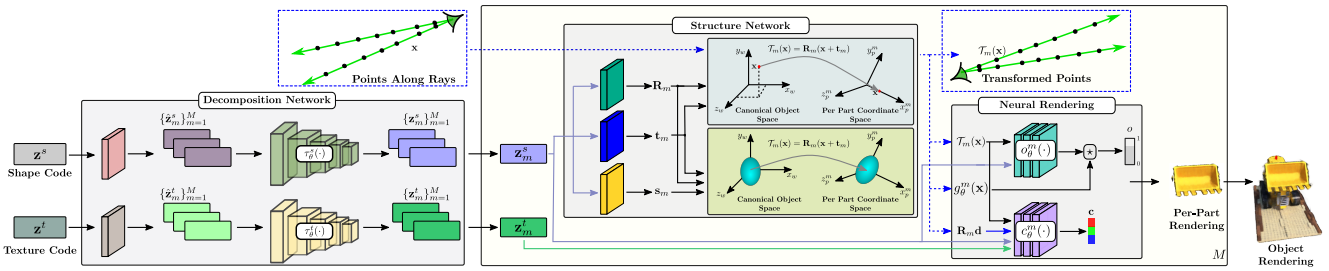

Figure 2. **Method Overview.** Our generative model is implemented as an auto-decoder and it comprises three main components: The *Decomposition Network* takes two object specific learnable embeddings $\{\mathbf{z}_s, \mathbf{z}_t\}$ that represent its shape and texture and maps them to a set of M latent codes that control the shape and texture of each part. First, we map $\mathbf{z}^s$ and $\mathbf{z}^t$ to M per-part embeddings $\{\hat{\mathbf{z}}^s_m\}_{m=1}^M$ and $\{\hat{\mathbf{z}}^t_m\}_{m=1}^M$ using M linear projections, which are then fed to two transformer encoders: τ^s_θ and τ^t_θ, that predict the final per-part shape and texture embeddings, $\{\mathbf{z}^s_m\}_{m=1}^M$ and $\{\mathbf{z}^t_m\}_{m=1}^M$. Next, the *Structure Network* maps the per-part shape feature representation $\mathbf{z}^s_m$ to a rotation matrix $\mathbf{R}_m$, a translation vector $\mathbf{t}_m$ and a scale vector $\mathbf{s}_m$ that define the coordinate system of the m-th part and its spatial extent. Finally, the *Neural Rendering* module takes the 3D points along each ray, transformed to the coordinate frame of its associated part, and maps them to an occupancy and a color value. We use plate notation to denote repetition over the M parts.

Unlike [8, 9, 25, 96] that are part agnostic, our model generates objects with part-level control, hence unlocking editing operations not previously possible. Our formulation is closely related to the compositional representation of [77] but has two important differences. First, we enforce a hard assignment between rays and parts ensuring that the color of each part is only determined by one NeRF, thus enabling local editing. Moreover, we model the shape and texture of each part separately, hence enabling more control.

Shape Editing using NeRFs: Our work falls into the category of shape editing approaches that operate on the radiance field [62, 115, 124]. Recent methods [115, 124] extract meshes from a pre-trained NeRF and rely on deformation techniques [22, 44, 60, 98] to guide the rendering process. Unlike our approach, these models are scene specific and cannot capture shape and texture variations across an object class. An alternative line of research explored using separate embeddings for capturing the shape and texture variations of 3D shapes [42, 62]. They demonstrated various editing operations such as color modifications and region removal. However, as they do not consider parts, they rely on heuristics for controlling what needs to be changed. Instead, our part-aware model provides more intuitive control, when editing a 3D shape. Also similar to our work, [80] uses multi-view videos and decomposes the object using a set of ellipsoids. However, [80] is scene-specific and cannot generate novel shapes. In the context of face editing [46, 100, 102], different models require video sequences and partial semantic masks [46], or posed images along with full semantic masks [100, 102]. In contrast, our work only requires posed images and 2D object masks at training.

Primitive-based Representations: Shape abstraction techniques represent 3D shapes using semantically consistent primitive arrangements across different instances in a class. This most often requires 3D supervision [17, 19, 26, 29, 50, 53, 54, 56, 69, 79, 85, 86, 97, 106, 128], although [118, 119] demonstrate that it is possible to learn only from images. Unlike our parts, geometric primitives are typically simple shapes such as cuboids [106], spheres [34] or superquadrics [85, 86]. Due to their simple shape parametrization these primitives cannot capture complex geometries. To address this, recent works propose to increase the number of primitives [17, 50] or represent shapes using a family of homeomorphic mappings [84], or a structured set of implicit functions [28]. Similar to [84] our parts can capture complex geometries, but as we parametrize them with locally defined NeRFs, we do not require explicit 3D supervision.

Part-based Shape Editing: Part-based generative models [34, 40, 55, 88, 111] can generate plausible 3D shapes [69, 70, 117] and perform various editing operations such as part mixing [58, 88, 121] and shape manipulation [34, 40]. Closely related to our work are [34, 40] which employ primitives, such as spheres [34] and 3D Gaussians [41], to enable shape editing by explicitly [34] or implicitly [40] transforming them. However, both require 3D supervision and can only alter attributes related to the shape of the object.

3. Method

Our goal is to design a 3D part-aware generative model that can be trained without explicit 3D supervision. Moreover, we want our generated shapes to be editable, namely to be able to make local changes on the shape and texture of specific parts of the object. To this end, we represent objects using M *locally defined* parts that are parametrized with a NeRF [68]. Defining NeRFs locally, *i.e.* in their own coordinate system, enables direct part-level control simply by applying transformations on the per-part coordinate system.

However, to achieve distinct, manipulable parts, we represent each part with a single NeRF. To enforce this, we introduce a *hard assignment* between rays and parts by asso-

ciating a ray with the first part it intersects (Fig. 3). Namely, the color of each ray is predicted from *a single NeRF*, thus preventing combinations of parts reasoning about the color of a ray. This ensures that editing one part, does not change the shape and appearance of the others (Fig. 1).

3.1. Neural Radiance Fields

Given a set of posed images of objects in a semantic class, each accompanied by an object mask, which is simply a binary image indicating whether each pixel is inside the object or not, we define $\mathcal{R}$, the complete set of rays from all views. For each ray $r = \{\mathbf{x}_0^r + t\mathbf{d}^r : t \geq 0\}$ with origin $\mathbf{x}_0^r$ and viewing direction $\mathbf{d}^r$, we denote $C(r) \in \mathbb{R}^3$ and $I(r) \in \{0, 1\}$ the color value of the RGB image and the binary value of the mask, respectively, at the corresponding pixel. Finally, we sample a set of N points $\mathcal{X}_r = \{\mathbf{x}_1^r, \ldots, \mathbf{x}_N^r\}$ along r, which are ordered by increasing distance from the origin $\mathbf{x}_0^r$, used for estimating the color along this ray using numerical quadrature [108].

Neural Radiance Fields: NeRFs [68] represent a scene as a continuous function, parametrized with an MLP, that maps a 3D point $\mathbf{x} \in \mathbb{R}^3$ and a viewing direction $\mathbf{d} \in \mathbb{S}^2$ into a *color* $\mathbf{c} \in \mathbb{R}^3$ and a volume *density* $\sigma \in \mathbb{R}^+$. Before passing the inputs $\mathbf{x}$ and $\mathbf{d}$ to the MLP, they are projected to a higher dimensional space by applying a fixed *positional encoding* [107] to each one of their elements. Given the predicted color and densities $\{\mathbf{c}_i^r, \sigma_i^r\}_{i=1}^N$ for the N sampled points $\mathcal{X}_r$ along ray r, its rendered color can be derived from

$$\hat{C}(r) = \sum_{i=1}^{N} \exp\Big(-\sum_{j<i} \sigma_j^r \delta_j^r\Big)(1 - \exp(-\sigma_i^r \delta_i^r))\mathbf{c}_i^r, \quad (1)$$

where δ_i^r is the distance between two adjacent samples along r. At training, the MLP is optimized by minimizing the error between observed and rendered images.

Alternatively, [82] propose predicting occupancies instead of densities, hence their rendering equation becomes

$$\hat{C}(r) = \sum_{i=1}^{N} o_i^r \prod_{j<i} \left(1 - o_j^r\right) \mathbf{c}_i^r, \quad (2)$$

where $o_i^r = 1 - \exp(-\sigma_i^r \delta_i^r)$ is the occupancy value at point $\mathbf{x}_i^r$ and $\mathbf{c}_i^r$ its color. Similar to [82] we also predict occupancy values, as this facilitates associating rays with parts, hence enabling part-level control, as discussed in Sec. 3.2.

3.2. Parts as Neural Radiance Fields

We represent a 3D object using M parts, where each part is parametrized as a NeRF. Note that we assume a fixed number of parts across all objects, namely the generated objects cannot have a variable number of parts. To learn the latent space of each NeRF, we follow [62] and condition it on two part-specific learnable latent codes: one for the shape and one for the texture, $\mathbf{z}_m^s, \mathbf{z}_m^t$. These codes are obtained from a per-object specific learnable embedding (see Sec. 3.3). Disentangling the shape from the texture allows modifying one property without affecting the other.

Moreover, as we are interested in editing specific parts of the object, we want to be able to modify the pose, size, and appearance of each part independently. This can be enforced by making sure that each NeRF receives geometric inputs in its own local coordinate system. To this end, we augment each part with: (i) an *affine transformation* $\mathcal{T}_m(\mathbf{x}) = \mathbf{R}_m(\mathbf{x} + \mathbf{t}_m)$ that maps a 3D point $\mathbf{x}$ to the local coordinate system of the m-th part, where $\mathbf{t}_m \in \mathbb{R}^3$ is the translation vector and $\mathbf{R}_m \in SO(3)$ is the rotation matrix and (ii) a *scale* vector $\mathbf{s}_m \in \mathbb{R}^3$, representing its spatial extent. All are obtained from the per-part shape code $\mathbf{z}_m^s$.

Part Representation: Each part is represented as a continuous function that maps a 3D point $\mathbf{x} \in \mathbb{R}^3$, a viewing direction $\mathbf{d} \in \mathbb{S}^2$, a shape code $\mathbf{z}^s \in \mathbb{R}^{L_s}$ and a texture code $\mathbf{z}^t \in \mathbb{R}^{L_t}$ into a color $\mathbf{c} \in \mathbb{R}^3$ and an occupancy value $o \in [0, 1]$. Similar to [82], we employ two separate networks: a *color network* c_θ and an *occupancy network* o_θ to predict the color and the occupancy value. Note that the occupancy value o is constrained to $[0, 1]$ with a sigmoid. More formally, each NeRF maps a 3D point $\mathbf{x}$ along a viewing direction $\mathbf{d}$ to a color $\mathbf{c}$ and an occupancy o as:

$$c_\theta^m(\mathbf{x}, \mathbf{d}) = c_\theta(\mathcal{T}_m(\mathbf{x}), \mathbf{R}_m\mathbf{d}, \mathbf{z}_m^s, \mathbf{z}_m^t) \quad (3)$$

$$o_\theta^m(\mathbf{x}) = o_\theta(\mathcal{T}_m(\mathbf{x}), \mathbf{z}_m^s). \quad (4)$$

Note that while we apply positional encoding on the inputs, we omit it from (3)+(4) to avoid notation clutter.

To enforce that each part only captures continuous regions of the object, we multiply its occupancy function with the occupancy function of an axis-aligned 3D ellipsoid centered at the origin of the coordinate system, with axis lengths given by the scale vector $\mathbf{s}_m$. This results in the following joint occupancy function for the m-th part

$$h_\theta^m(\mathbf{x}) = o_\theta^m(\mathbf{x}) g_\theta^m(\mathbf{x}), \quad (5)$$

where $g_\theta^m(\mathbf{x}) = g(T_m(\mathbf{x}), \mathbf{s}_m)$ denotes the occupancy function of the m-th ellipsoid that is simply

$$g(\mathbf{x}, \mathbf{s}) = \sigma\left(\beta\left(1 - \left\|\mathrm{diag}(\mathbf{s})^{-1}\mathbf{x}\right\|^2\right)\right), \quad (6)$$

where $\sigma(\cdot)$ is the sigmoid function and β controls the sharpness of the transition. To estimate $g_\theta^m(\mathbf{x})$, we first transform $\mathbf{x}$ to the coordinate frame of the m-th part. This ensures that any transformation of the part, also transforms its ellipsoid.

Part Rendering: The rendering equation of the m-th part, given a set of N sampled points along ray r now becomes

$$\hat{C}_m(r) = \sum_{i=1}^{N} h_\theta^m(\mathbf{x}_i^r) \prod_{j<i} \left(1 - h_\theta^m(\mathbf{x}_i^r)\right) c_\theta^m(\mathbf{x}_i^r, \mathbf{d}^r). \quad (7)$$

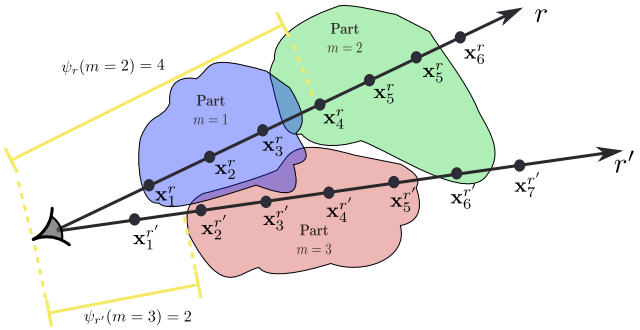

Figure 3. **Ray-Part Association.** We illustrate the hard assignment between rays and parts in a 2D example with 3 parts and two rays r and r'. Since the association between rays and parts is determined based on the first part that a ray interests, the associations that emerge from (9) are $\mathcal{R}_1 = \{r\}, \mathcal{R}_2 = \{\emptyset\}$ and $\mathcal{R}_3 = \{r'\}$.

Object Rendering: To ensure distinct, manipulable parts, we introduce a *hard assignment* between rays and parts, by associating a ray with the first part it intersects. Given the ordered set of points $\mathcal{X}_r$ sampled along ray r, we define the index of the first point inside the part that r intersects as

$$\psi_r(m) = \min\left\{i \in \{1,\dots,N\} \,:\, h_\theta^m(\mathbf{x}_i^r) \geq \tau\right\}, \quad (8)$$

where τ is a threshold used to determine whether a point is *inside* the m-th part or not. This is illustrated in Fig. 3. We can now define the set of rays $\mathcal{R}_m$ associated with the m-th part, as the set of rays that first intersect with it, namely:

$$\mathcal{R}_m = \left\{r \in \mathcal{R} \,:\, m = \underset{k\in\{0\dots M\}}{\operatorname{argmin}} \; \psi_r(k)\right\}. \quad (9)$$

Using the assignment of rays to parts, we define the rendering equation for the entire object as

$$\hat{C}(r) = \sum_{m=1}^{M} \mathbb{1}_{r\in\mathcal{R}_m} \hat{C}_m(r). \quad (10)$$

Namely, we use the m-th NeRF to render ray r if it is assigned to the m-th part. If a ray is not associated with any part its color is black. The hard ray-part assignment is an essential property that ensures that editing one part does not alter the appearance of other parts (see Fig. 5).

3.3. Network Architecture

We implement PartNeRF using an auto-decoder [5, 83]. The input to our model is two learnable embeddings $\mathbf{z}^s, \mathbf{z}^t$, per training sample, that represent its shape and texture. Our network consists of three components: (i) the *decomposition network* that maps $\mathbf{z}^s$ and $\mathbf{z}^t$ to M latent codes that control the per-part shape and texture, (ii) the *structure network* that predicts the per-part pose and scale and (iii) the *neural rendering network* that renders an image using M NeRFs. The overall architecture is illustrated in Fig. 2.

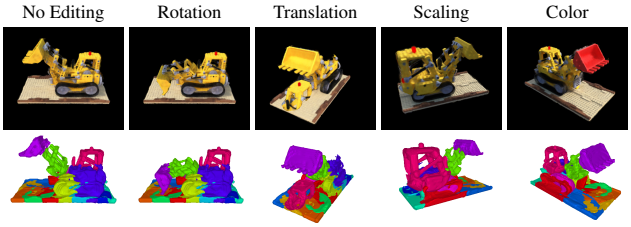

Figure 4. **Scene-Specific Editing**. We show renderings (top row) and part-based geometries (bottom row) for a tractor from novel views across various editing operations: *rotating* the bucket downwards, *translating* the cockpit to the floor , performing *isotropic scaling* of the cockpit and *coloring* the bucket red.

Decomposition Network: The decomposition network takes the two object specific embeddings $\mathbf{z}^s, \mathbf{z}^t \in \mathbb{R}^{L_d}$ and maps them to M per-part embeddings of the same dimensionality, L_d, using M linear projections $f_\theta(\cdot)$. Subsequently, these embeddings are mapped to part-specific latent codes using two multi-head attention transformers, τ_θ^s and τ_θ^t without positional encoding [107] as follows

$$\{\mathbf{z}_m^s\}_{m=1}^M = \tau_\theta^s(f_\theta(\mathbf{z}^s)) \quad (11)$$

$$\{\mathbf{z}_m^t\}_{m=1}^M = \tau_\theta^t(f_\theta(\mathbf{z}^t)). \quad (12)$$

$\{\mathbf{z}_m^s\}_{m=1}^M$ and $\{\mathbf{z}_m^t\}_{m=1}^M$ are the latent codes that control the shape and texture of each part respectively.

Structure Network: The structure network s_θ maps the shape latent code $\mathbf{z}_m^s$ to a translation vector $\mathbf{t}_m$, a rotation matrix $\mathbf{R}_m$ and a scale vector $\mathbf{s}_m$ with

$$\{\mathbf{t}_m, \mathbf{R}_m, \mathbf{s}_m\} = s_\theta(\mathbf{z}_m^s) \quad (13)$$

an MLP shared across parts. Similar to [85], we parametrize $\mathbf{R}_m$ using quaternions [33]. As discussed in Sec. 3.2, we determine the set of rays $\mathcal{R}_m$ that are assigned to each part m from (9) and transform them to its coordinate system.

Neural Rendering: Given the transformed points to the per-part coordinate system, we predict colors and occupancies with (5)+(3). Next, we perform volumetric rendering using (7) and render the object using M NeRFs with (10).

3.4. Training

Our optimization objective $\mathcal{L}$ is the sum over six terms combined with two regularizers on the shape and texture embeddings $\mathbf{z}^s, \mathbf{z}^t$, namely

$$\mathcal{L} = \mathcal{L}_{rgb}(\mathcal{R}) + \mathcal{L}_{mask}(\mathcal{R}) + \mathcal{L}_{occ}(\mathcal{R}) + \mathcal{L}_{cov}(\mathcal{R}) + \mathcal{L}_{overlap}(\mathcal{R}) + \mathcal{L}_{control} + \|\mathbf{z}^s\|_2 + \left\|\mathbf{z}^t\right\|_2. \quad (14)$$

As supervision, we use the observed RGB color $C(r) \in \mathbb{R}^3$ and the object mask $I(r) \in \{0, 1\}$ for each ray $r \in \mathcal{R}$. We also associate r with a binary label $\ell_r = I(r)$. Namely, we label a ray r as *inside*, if $\ell_r = 1$ and *outside* if $\ell_r = 0$.

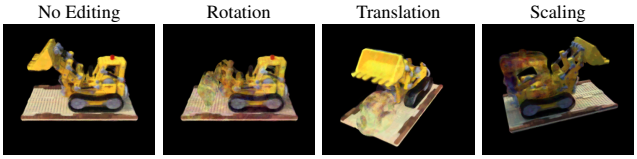

Figure 5. **Soft Ray-Part Assignment**. We demonstrate that enforcing a soft ray-part assignment results in parts that do not preserve their texture across transformations.

Reconstruction Loss: We measure the error between the observed $C(r)$ and the rendered $\hat{C}(r)$ color for the ray r as

$$\mathcal{L}_{rgb}(\mathcal{R}) = \sum_{r\in\mathcal{R}} \|\hat{C}(r) - C(r)\|_2^2. \tag{15}$$

Mask Loss: Likewise, we measure the squared error between the observed $I(r)$ and the rendered $\hat{I}(r)$ pixel value of the object mask for ray r as

$$\mathcal{L}_{mask}(\mathcal{R}) = \sum_{r\in\mathcal{R}} \|\hat{I}(r) - I(r)\|_2^2. \tag{16}$$

Note that $\hat{I}(r)$ can be derived from (10)+(7) simply by omitting the multiplication with the predicted color, namely

$$\hat{I}(r) = \sum_{m=1}^{M} \mathbb{1}_{r\in\mathcal{R}_m} \sum_{i=1}^{N} h_\theta^m(\mathbf{x}_i^r) \prod_{j<i} \left(1 - h_\theta^m(\mathbf{x}_j^r)\right). \tag{17}$$

Occupancy Loss: This loss makes sure that the generated parts do not occupy empty space. To this end, we employ a binary cross-entropy classification loss $\mathcal{L}_{ce}(\cdot;\cdot)$ between the predicted and the target labels for all rays

$$\mathcal{L}_{occ}(\mathcal{R}) = \frac{1}{|\mathcal{R}|} \sum_{r\in\mathcal{R}} \Big(\mathcal{L}_{ce}(\hat{\ell}_r, \ell_r) + \mathcal{L}_{ce}(\tilde{\ell}_r, \ell_r)\Big), \tag{18}$$

where $\hat{\ell}_r$ and $\tilde{\ell}_r$ are the predicted labels along ray r based on the predicted occupancies and ellipsoid occupancies respectively. Intuitively, we consider a ray r to be inside the object if it is inside at least one part. In turn, in order for r to be inside a part it suffices if at least one point along the ray $\mathcal{X}_r$ is inside this part. This can be expressed as

$$\hat{\ell}_r = \max_{m\in\{1\dots M\}} \max_{\mathbf{x}_i^r\in\mathcal{X}_r} h_\theta^m(\mathbf{x}_i^r) \tag{19}$$

$$\tilde{\ell}_r = \max_{m\in\{1\dots M\}} \max_{\mathbf{x}_i^r\in\mathcal{X}_r} g_\theta^m(\mathbf{x}_i^r). \tag{20}$$

Coverage Loss: This loss ensures that the parts cover the object, thus preventing degenerate arrangements with small parts or parts outside the object. To implement this, we encourage parts to contain at least k *inside* rays. This can be

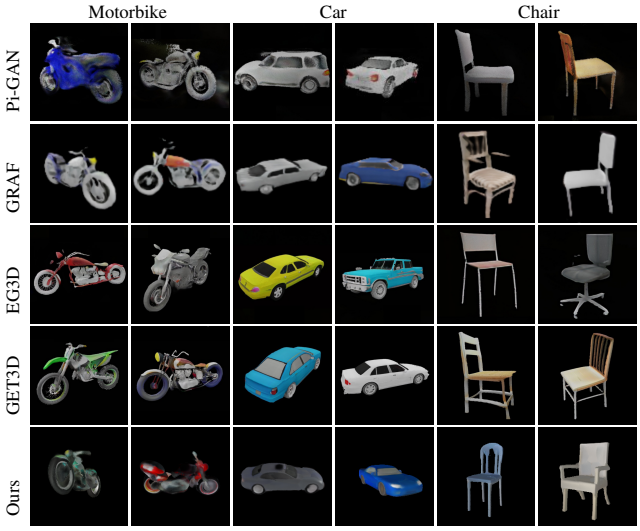

Figure 6. **Shape Generation**. We show two randomly generated samples per category using our model and 3D generative models that are part agnostic but can generate textured meshes.

Method	Rendering	MMD-CD (↓)			COV-CD (%, ↑)		
		Motorbike	Car	Chair	Motorbike	Car	Chair
GET3D	Differentiable	1.72	0.71	3.72	67.12	58.39	69.91
Pi-GAN	Volumetric	21.80	25.54	6.65	6.85	0.55	39.65
GRAF	Volumetric	2.40	10.63	6.80	50.68	1.57	39.28
EG3D	Volumetric	2.21	**0.72**	4.72	34.25	**49.52**	50.14
Ours	Volumetric	**1.68**	1.74	**4.42**	**56.06**	21.10	**67.20**

Table 1. **Comparison with 3D Generative Models.** We measure MMD-CD (↓) and COV-CD (↑). Note that none of these baselines considers parts nor allows any part-level shape editing.

expressed as a binary cross-entropy loss between the predicted per-part and target labels for all rays in $\mathcal{R}_m^k$,

$$\mathcal{L}_{cov}(\mathcal{R}) = \frac{1}{M} \sum_{m=1}^{M} \sum_{r\in\mathcal{R}_m^k} \mathcal{L}_{ce}(\hat{\ell}_r^m, \ell_r), \tag{21}$$

where $\hat{\ell}_r^m = \max_{\mathbf{x}_i^r\in\mathcal{X}_r} h_\theta^m(\mathbf{x}_i^r)$ the predicted label for ray r w.r.t. part m and $\mathcal{R}_m^k$ the set of the k inside rays with the greatest predicted occupancy values for this part.

Overlapping Loss: To encourage the generated parts to capture different regions of the object, we penalize rays that are inside of more than λ parts as follows

$$\mathcal{L}_{overlap}(\mathcal{R}) = \frac{1}{|\mathcal{R}|} \sum_{r\in\mathcal{R}} \max\Big(0, \sum_{m=1}^{M} \hat{\ell}_r^m - \lambda\Big). \tag{22}$$

Control Loss: To ensure uniform control across the shape, we want parts with comparable volumes. We implement this loss on the volumes $\mathcal{V}(\cdot)$ of the ellipsoids, as follows:

$$\mathcal{L}_{control} = \frac{1}{M(M-1)} \sum_{i=1}^{M} \sum_{j=1}^{i} |\mathcal{V}(\mathbf{s}_i) - \mathcal{V}(\mathbf{s}_j)|. \tag{23}$$

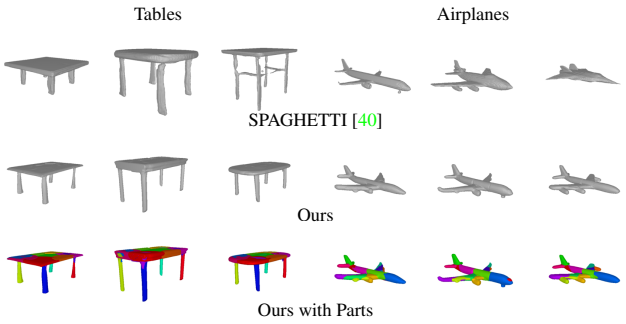

Figure 7. **Shape Generation**. We compare our model with [40] and show three randomly generated samples per category.

Method	Supervision	MMD-CD (↓)		COV-CD (%, ↑)	
		Airplane	Table	Airplane	Table
DualSDF	3D Shapes	4.20	12.30	25.00	36.30
SPAGHETTI	3D Shapes	2.40	5.90	35.00	**47.80**
Ours	Multi-view	**1.37**	**4.48**	**37.90**	40.60

Table 2. **Comparison with Part-based Generative Models.** We measure MMD-CD (↓) and the COV-CD (↑). Unlike our model, both [34, 40] require 3D supervision during training.

4. Experimental Evaluation

We provide an extensive evaluation of PartNeRF comparing it to relevant baselines in terms of the realism and diversity of the generated shapes. We also showcase several editing operations of our model on multiple object categories. Additional results, ablations and implementation details are provided in the supplementary.

Datasets: We use five ShapeNet [10] categories: *Motorbike, Chair, Table, Airplane and Car*. To render our training data, we randomly sample camera poses from the upper hemisphere of each shape and render images at 256^2 resolution as in [25]. For the *Car, Table, Airplane and Chair*, we use 24 random views, while for Motorbike we use 100. To ensure fair comparison, we use the train-test splits of [25] for the *Motorbike, Car, Chair* object categories and the train-test splits of [40] for the *Airplane, Table* category. In all experiments, we perform category specific training and set $M = 16$. Finally, we showcase the scene-specific editing capabilities of our model on the Lego tractor [68].

Baselines: We compare our model with several NeRF-based models: GRAF [96], Pi-GAN [9], EG3D [8] and the concurrent GET3D [25] that relies on differentiable rendering. Unlike our model, none of the above considers parts. We also compare with the part-based DualSDF [34] and SPAGHETTI [40] that require 3D supervision.

Metrics: We report the Coverage (COV) and the Minimum Matching Distance (MMD) [1] using Chamfer-L_2 distance. MMD measures how likely it is that a generated shape looks like a test shape. COV measures how many shape variations are covered by the generated shapes.

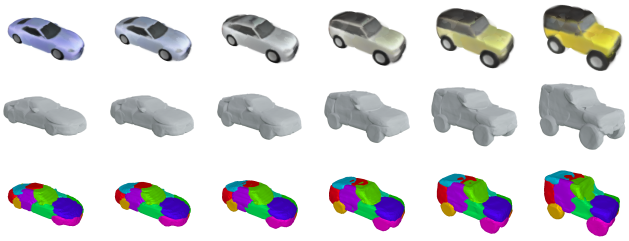

Figure 8. **Shape Interpolation**. From left to right, we interpolate between the geometry and texture latent codes of the two shapes.

4.1. Scene-Specific Shape Editing

We train PartNeRF on the tractor scene [68], using 200 training views. The results are shown in Fig. 4. Initially, we select the part that corresponds to the bucket of the tractor and apply a rotation, such that the bucket is facing downwards. Similarly, we select the cockpit and move it to a new location on the floor, by adding a displacement to the part's translation vector. We then apply a non-rigid transformation on the cockpit, scaling it uniformly across all axes, by multiplying the part's rotation with an isometric scale matrix. Note that during all transformations the rest of the shape (geometry, parts and texture) does not alter. Finally, we alter the color of the bucket by explicitly setting the predicted color of its associated NeRF to red.

Soft Ray-Part Assignment: To investigate the impact of our hard ray-part assignment, we train a variant of our model without, namely the color of a ray can be determined from multiple NeRFs. We note that when we apply a transformation on a part of the object, also the color of other parts changes, as illustrated in Fig. 5. Moreover, note that with this variant of our model it is not possible to change the color of specific parts of the generated object, as in Fig. 4.

4.2. Shape Generation

In Tab. 1, we compare the quality of our generations with NeRF-based generative models and observe that it outperforms existing approaches in terms of MMD and COV on Motorbikes and Chairs, while being better than [9, 96] also for Cars. Furthermore, it performs on par with the concurrent work of [25] that relies on a highly optimized differentiable graphics renderer [51] and requires training for approximately 16 GPU days (8 A100 for 2 days). Instead our model employs a simpler volumetric renderer and training takes approximately 5 GPU days (1 RTX 3090). Compared to [9, 96], our generations are sharper with more crisp colors (see Fig. 6). However, compared to [8, 25] that employ tri-plane representations and hence can generate high-resolution textures, our textures are less detailed.

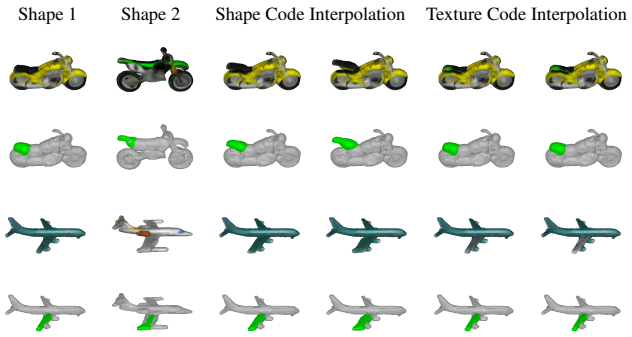

Figure 9. **Part-Level Interpolation**. From left to right, we show interpolations for the motorbike's saddle and the airplane's right wing (colored in green). In the 3rd and 4th columns, we interpolate between the shape codes of the two parts, whereas in the 5th and 6th we interpolate between the texture codes of the two parts.

We also compare PartNeRF with state-of-the-art part-based approaches that require 3D supervision. While our model is trained with posed images and object masks, it consistently generates plausible 3D geometries (see Fig. 7). This is also validated quantitatively in Tab. 2, where we observe that our model outperforms both [34,40] on Airplanes while being better in terms of MMD for Tables.

Shape Interpolation: In Fig. 8, we show interpolations between the shape and texture codes of two cars. We observe that our model smoothly interpolates between two shapes, while preserving the shape and the part-based structure.

Part-level Interpolation: Fig. 9 shows part-level interpolations, where we select the saddle part for the motorbikes and the right wing for the airplanes and linearly interpolate its shape and texture codes. We note that when we interpolate the shape codes, the geometry changes, while the texture remains the same. In contrast, when we interpolate the texture codes, the geometry remains unchanged and only the part texture changes smoothly. Across all interpolations our model consistently generates realistic shapes.

4.3. Shape and Texture Editing

Shape Mixing: Starting from two shapes, the task is to select and combine parts in a meaningful way. As shown in Fig. 10, we consider two types of mixing operations: *geometry* and *texture* mixing. In geometry mixing, we combine parts from two objects and generate a new one, whose texture is determined by the texture codes from one of the two, while shape codes are taken from different parts of the two objects (third column). In texture mixing, we mix the shapes only in terms of texture, while the shape is determined by the shape codes of one object (fourth column). We also combine both mixing modes. PartNeRF consistently generates plausible shapes across all editing operations.

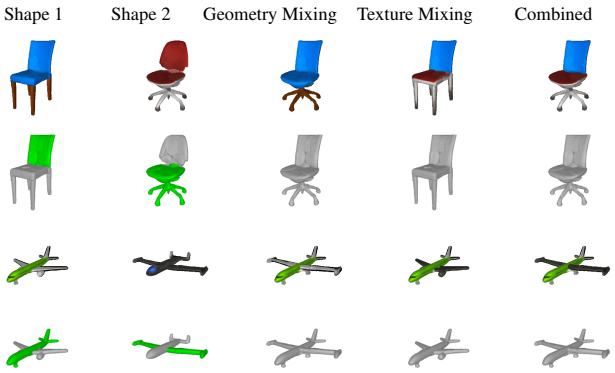

Figure 10. **Shape Mixing**. We mix parts (colored in green) from two shapes and show *geometry* (3rd column), *texture* (4th column), and combined geometry and texture mixing (5th column).

Shape Editing: Our method allows several shape editing operations on the part level, such as applying rigid and non-rigid transformations and inserting or removing parts (see Fig. 1). Affine transformations are directly applied on the rotation and translation vectors that define the per-part coordinate frame. To remove a part, it suffices to ignore its associated NeRF in the rendering process. Likewise, adding a part amounts to incorporating its NeRF during rendering.

5. Conclusion

In this paper, we introduce PartNeRF, the first part-aware generative model that parametrizes parts as NeRFs. As our work considers the decomposition of objects into parts, it enables intuitive part-level control and several editing operations not previously possible. Furthermore, it is trained without explicit 3D supervision, using only posed images and object masks. Our experiments showcase the ability of our model to generate plausible 3D shapes with texture. Moreover, we demonstrate several editing operations both on the texture and the shape of the generated object. In future work, we plan to investigate incorporating more complex representations such as triplanes [8], or using differentiable rendering techniques [25] in order to better represent the object's texture. Another exciting direction for future research is extending our model to moving objects.

Acknowledgments

Konstantinos Tertikas and Ioannis Emiris are supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 860843. This work was supported by an ARL grant W911NF-21-2-0104 and an Adobe research gift. Despoina Paschalidou is supported by the Swiss National Science Foundation under grant number P500PT_206946. Leonidas Guibas is supported by a Vannevar Bush Faculty Fellowship.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. Learning representations and generative models for 3d point clouds. In *Proc. of the International Conf. on Machine learning (ICML)*, 2018. 7

[2] Matan Atzmon, Niv Haim, Lior Yariv, Ofer Israelov, Haggai Maron, and Yaron Lipman. Controlling neural level sets. In *Advances in Neural Information Processing Systems (NIPS)*, 2019. 2

[3] Matan Atzmon and Yaron Lipman. SAL: sign agnostic learning of shapes from raw data. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2562–2571, 2020. 2

[4] Sai Bi, Zexiang Xu, Pratul P. Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Milos Hasan, Yannick Hold-Geoffroy, David J. Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv.org*, 2020. 2

[5] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space of generative networks. In *Proc. of the International Conf. on Machine learning (ICML)*, 2018. 2, 5

[6] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T. Barron, Ce Liu, and Hendrik P. A. Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2

[7] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2019. 2

[8] 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 *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3, 7, 8

[9] 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 *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3, 7

[10] 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. *arXiv.org*, 1512.03012, 2015. 7

[11] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser-Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2

[12] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[13] Zezhou Cheng, Menglei Chai, Jian Ren, Hsin-Ying Lee, Kyle Olszewski, Zeng Huang, Subhransu Maji, and Sergey Tulyakov. Cross-modal 3d shape generation and manipulation. *arXiv.org*, 2022. 2

[14] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[15] 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 *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[16] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[17] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. Cvxnets: Learnable convex decomposition. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[18] Yu Deng, Jiaolong Yang, and Xin Tong. Deformed implicit field: Modeling 3d shapes with learned dense correspondence. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 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 *Advances in Neural Information Processing Systems (NIPS)*, 2019. 3

[20] Terrance DeVries, Miguel Ángel Bautista, Nitish Srivastava, Graham W. Taylor, and Joshua M. Susskind. Unconstrained scene generation with locally conditioned radiance fields. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2

[21] Tim Elsner, Moritz Ibing, Victor Czech, Julius Nehring-Wirxel, and Leif Kobbelt. Intuitive shape editing in latent space. *arXiv.org*, 2021. 2

[22] Michael S Floater. Mean value coordinates. *Computer Aided Geometric Design*, 2003. 3

[23] Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2017. 2

[24] Jun Gao, Wenzheng Chen, Tommy Xiang, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Learning deformable tetrahedral meshes for 3d reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[25] 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. 2, 3, 7, 8

[26] Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao Zhang. SDM-NET: deep generative network for structured deformable mesh. In *ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia (SIGGRAPH Asia)*, 2019. 3

[27] Stephan J. Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien P. C. Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2

[28] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas A. Funkhouser. Local deep implicit functions for 3d shape. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[29] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 3

[30] 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 *Advances in Neural Information Processing Systems (NIPS)*, 2014. 2

[31] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *Proc. of the International Conf. on Machine learning (ICML)*, 2020. 2

[32] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d aware generator for high-resolution image synthesis. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2022. 2

[33] William Rowan Hamilton. Xi. on quaternions; or on a new system of imaginaries in algebra. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 33(219):58–60, 1848. 5

[34] Zekun Hao, Hadar Averbuch-Elor, Noah Snavely, and Serge J. Belongie. Dualsdf: Semantic shape manipulation using a two-level representation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 3, 7, 8

[35] Zekun Hao, Arun Mallya, Serge J. Belongie, and Ming-Yu Liu. Gancraft: Unsupervised 3d neural rendering of minecraft worlds. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2

[36] Paul Henderson and Vittorio Ferrari. Learning single-image 3d reconstruction by generative modelling of shape, pose and shading. *International Journal of Computer Vision (IJCV)*, 2019. 2

[37] Paul Henderson and Christoph H. Lampert. Unsupervised object-centric video generation and decomposition in 3d. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[38] Philipp Henzler, Niloy J Mitra, , and Tobias Ritschel. Escaping plato's cave: 3d shape from adversarial rendering. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

[39] Philipp Henzler, Jeremy Reizenstein, Patrick Labatut, Roman Shapovalov, Tobias Ritschel, Andrea Vedaldi, and David Novotný. Unsupervised learning of 3d object categories from videos in the wild. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[40] 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, 2, 3, 7, 8

[41] Amir Hertz, Or Perel, Raja Giryes, Olga Sorkine-Hornung, and Daniel Cohen-Or. Spaghetti: Editing implicit shapes through part aware generation. *arXiv.org*, 2022. 3

[42] Wonbong Jang and Lourdes Agapito. Codenerf: Disentangled neural radiance fields for object categories. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2, 3

[43] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas Funkhouser. Local implicit grid representations for 3d scenes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[44] Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. Harmonic coordinates for character articulation. *ACM Trans. on Graphics*, 2007. 3

[45] James T. Kajiya and Brian Von Herzen. Ray tracing volume densities. In *ACM Trans. on Graphics*, 1984. 2

[46] Kacper Kania, Kwang Moo Yi, Marek Kowalski, Tomasz Trzciński, and Andrea Tagliasacchi. Conerf: Controllable neural radiance fields. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[47] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2018. 2

[48] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2

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

[50] Yuki Kawana, Yusuke Mukuta, and Tatsuya Harada. Neural star domain as primitive representation. In *Advances in Neural Information Processing Systems (NIPS)*, 2020. 3

[51] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Trans. on Graphics*, 2020. 7

[52] Verica Lazova, Vladimir Guzov, Kyle Olszewski, Sergey Tulyakov, and Gerard Pons-Moll. Control-nerf: Editable feature volumes for scene rendering and manipulation. *arXiv.org*, 2022. 2

[53] 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. 3

[54] Lingxiao Li, Minhyuk Sung, Anastasia Dubrovina, Li Yi, and Leonidas Guibas. Supervised fitting of geometric primitives to 3d point clouds. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[55] 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. 2, 3

[56] Yichen Li, Kaichun Mo, Lin Shao, Minhyuk Sung, and Leonidas J. Guibas. Learning 3d part assembly from a single image. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 3

[57] Yiyi Liao, Katja Schwarz, Lars M. Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[58] Connor Z. Lin, Niloy J. Mitra, Gordon Wetzstein, Leonidas J. Guibas, and Paul Guerrero. Neuform: Adaptive overfitting for neural shape editing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 3

[59] David B. Lindell, Julien N. P. Martel, and Gordon Wetzstein. Autoint: Automatic integration for fast neural volume rendering. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[60] Yaron Lipman, David Levin, and Daniel Cohen-Or. Green coordinates. *ACM Trans. on Graphics*, 2008. 3

[61] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[62] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2, 3, 4

[63] Sebastian Lunz, Yingzhen Li, Andrew W. Fitzgibbon, and Nate Kushman. Inverse graphics gan: Learning to generate 3d shapes from unstructured 2d data. *arXiv.org*, 2020. 2

[64] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[65] Quan Meng, Anpei Chen, Haimin Luo, Minye Wu, Hao Su, Lan Xu, Xuming He, and Jingyi Yu. Gnerf: Gan-based neural radiance field without posed camera. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2

[66] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[67] Mateusz Michalkiewicz, Jhony K Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Implicit surface representations as layers in neural networks. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

[68] 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 *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 2, 3, 4, 7

[69] 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. 3

[70] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy J. Mitra, and Leonidas J. Guibas. Structedit: Learning structural shape variations. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3

[71] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. on Graphics*, 2022. 2

[72] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

[73] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

[74] Thu Nguyen-Phuoc, Christian Richardt, Long Mai, Yong-Liang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *arXiv.org*, 2020. 2

[75] Thu Nguyen-Phuoc, Christian Richardt, Long Mai, Yong-Liang Yang, and Niloy J. Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[76] Michael Niemeyer and Andreas Geiger. CAMPARI: camera-aware decomposed generative neural radiance fields. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2021. 2

[77] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3

[78] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[79] Chengjie Niu, Jun Li, and Kai Xu. Im2struct: Recovering 3d shape structure from a single RGB image. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[80] Atsuhiro Noguchi, Umar Iqbal, Jonathan Tremblay, Tatsuya Harada, and Orazio Gallo. Watch it move: Unsupervised discovery of 3d joints for re-posing of articulated objects. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[81] Michael Oechsle, Michael Niemeyer, Christian Reiser, Lars Mescheder, Thilo Strauss, and Andreas Geiger. Learning implicit surface light fields. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2020. 2

[82] Michael Oechsle, Songyou Peng, and Andreas Geiger. UNISURF: unifying neural implicit surfaces and radiance

fields for multi-view reconstruction. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2, 4

[83] Jeong Joon Park, Peter Florence, Julian Straub, Richard A. Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 5

[84] Despoina Paschalidou, Angelos Katharopoulos, Andreas Geiger, and Sanja Fidler. Neural parts: Learning expressive 3d shape abstractions with invertible neural networks. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[85] Despoina Paschalidou, Ali Osman Ulusoy, and Andreas Geiger. Superquadrics revisited: Learning 3d shape parsing beyond cuboids. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3, 5

[86] Despoina Paschalidou, Luc van Gool, and Andreas Geiger. Learning unsupervised hierarchical part decomposition of 3d objects from a single rgb image. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[87] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 2

[88] Dmitry Petrov, Matheus Gadelha, Radomír Mech, and Evangelos Kalogerakis. ANISE: assembly-based neural implicit surface reconstruction. *arXiv.org*, 2022. 2, 3

[89] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. Derf: Decomposed radiance fields. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[90] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2

[91] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotný. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2

[92] Edoardo Remelli, Artem Lukoianov, Stephan R. Richter, Benoît Guillard, Timur M. Bagautdinov, Pierre Baqué, and Pascal Fua. Meshsdf: Differentiable iso-surface extraction. 2020. 2

[93] 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 *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

[94] Shunsuke Saito, Tomas Simon, Jason M. Saragih, and Hanbyul Joo. Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[95] Thorsten-Walther Schmidt, Fabio Pellacini, Derek Nowrouzezahrai, Wojciech Jarosz, and Carsten Dachsbacher. State of the art in artistic editing of appearance, lighting and material. *Comput. Graph. Forum*, 2016. 2

[96] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2, 3, 7

[97] Gopal Sharma, Difan Liu, Subhransu Maji, Evangelos Kalogerakis, Siddhartha Chaudhuri, and Radomír Mech. Parsenet: A parametric surface fitting network for 3d point clouds. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 3

[98] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Eurographics Symposium on Geometry Processing (SGP)*, 2007. 3

[99] Pratul P. Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T. Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[100] Jingxiang Sun, Xuan Wang, Yichun Shi, Lizhen Wang, Jue Wang, and Yebin Liu. Ide-3d: Interactive disentangled editing for high-resolution 3d-aware portrait synthesis. *arXiv.org*, 2022. 3

[101] Jingxiang Sun, Xuan Wang, Yong Zhang, Xiaoyu Li, Qi Zhang, Yebin Liu, and Jue Wang. Fenerf: Face editing in neural radiance fields. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[102] Jingxiang Sun, Xuan Wang, Yong Zhang, Xiaoyu Li, Qi Zhang, Yebin Liu, and Jue Wang. Fenerf: Face editing in neural radiance fields. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[103] Matthew Tancik, Ben Mildenhall, Terrance Wang, Divi Schmidt, Pratul P. Srinivasan, Jonathan T. Barron, and Ren Ng. Learned initializations for optimizing coordinate-based neural representations. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[104] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul P. Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, Tomas Simon, Christian Theobalt, Matthias Nießner, Jonathan T. Barron, Gordon Wetzstein, Michael Zollhöfer, and Vladislav Golyanik. Advances in neural rendering. *Computer Graphics Forum*, 2022. 2

[105] Alex Trevithick and Bo Yang. GRF: learning a general radiance field for 3d representation and rendering. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2

[106] Shubham Tulsiani, Hao Su, Leonidas J. Guibas, Alexei A. Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[107] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, 2017. 4, 5

[108] John Wallis. *Arithmetica Infinitorum*. 1656. 4

[109] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural

implicit surfaces by volume rendering for multi-view reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2

[110] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P. Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas A. Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[111] Rundi Wu, Yixin Zhuang, Kai Xu, Hao Zhang, and Baoquan Chen. PQ-NET: A generative part seq2seq network for 3d shapes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3

[112] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. In *Computer Graphics Forum*, 2022. 2

[113] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomír Mech, and Ulrich Neumann. DISN: deep implicit surface network for high-quality single-view 3d reconstruction. In *Advances in Neural Information Processing Systems (NIPS)*, 2019. 2

[114] Yang Xue, Yuheng Li, Krishna Kumar Singh, and Yong Jae Lee. GIRAFFE HD: A high-resolution 3d-aware generative model. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[115] Bangbang Yang, Chong Bao, Junyi Zeng, Hujun Bao, Yinda Zhang, Zhaopeng Cui, and Guofeng Zhang. Neumesh: Learning disentangled neural mesh-based implicit field for geometry and texture editing. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2022. 2, 3

[116] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2

[117] Jie Yang, Kaichun Mo, Yu-Kun Lai, Leonidas J. Guibas, and Lin Gao. Dsm-net: Disentangled structured mesh net for controllable generation of fine geometry. *arXiv.org*, 2020. 2, 3

[118] Chun-Han Yao, Wei-Chih Hung, Varun Jampani, and Ming-Hsuan Yang. Discovering 3d parts from image collections. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 3

[119] 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. *arXiv.org*, abs/2207.03434, 2022. 3

[120] Lior Yariv, Matan Atzmon, and Yaron Lipman. Universal differentiable renderer for implicit neural representations. *arXiv.org*, 2003.09852, 2020. 2

[121] Kangxue Yin, Zhiqin Chen, Siddhartha Chaudhuri, Matthew Fisher, Vladimir G Kim, and Hao Zhang. Coalesce: Component assembly by learning to synthesize connections. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2020. 3

[122] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2021. 2

[123] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[124] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. Nerf-editing: Geometry editing of neural radiance fields. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3

[125] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv.org*, 2020. 2

[126] 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 *Proc. of the International Conf. on Learning Representations (ICLR)*, 2021. 2

[127] Peng Zhou, Lingxi Xie, Bingbing Ni, and Qi Tian. Cips-3d: A 3d-aware generator of gans based on conditionally-independent pixel synthesis. *arXiv.org*, abs/2110.09788, 2021. 2

[128] Chuhang Zou, Ersin Yumer, Jimei Yang, Duygu Ceylan, and Derek Hoiem. 3d-prnn: Generating shape primitives with recurrent neural networks. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2017. 3