GaussianCube: Structuring Gaussian Splatting using Optimal Transport for 3D Generative Modeling

Bowen Zhang[1*] **Yiji Cheng**[2*] **Jiaolong Yang**[3] **Chunyu Wang**[3]
Feng Zhao[1] **Yansong Tang**[2] **Dong Chen**[3] **Baining Guo**[3]
[1]University of Science and Technology of China [2]Tsinghua University [3]Microsoft Research Asia

Abstract

3D Gaussian Splatting (GS) have achieved considerable improvement over Neural Radiance Fields in terms of 3D fitting fidelity and rendering speed. However, this unstructured representation with scattered Gaussians poses a significant challenge for generative modeling. To address the problem, we introduce GaussianCube, a structured GS representation that is both powerful and efficient for generative modeling. We achieve this by first proposing a modified densification-constrained GS fitting algorithm which can yield high-quality fitting results using a fixed number of free Gaussians, and then re-arranging the Gaussians into a predefined voxel grid via Optimal Transport. The structured grid representation allows us to use standard 3D U-Net as our backbone in diffusion generative modeling without elaborate designs. Extensive experiments conducted on ShapeNet and OmniObject3D show that our model achieves state-of-the-art generation results both qualitatively and quantitatively, underscoring the potential of GaussianCube as a powerful and versatile 3D representation. Project page: https://gaussiancube.github.io/.

1 Introduction

Recent advancements in generative modeling [23, 17, 35, 14, 64, 27] have led to significant progress in 3D content creation [53, 33, 4, 49, 41, 6, 16]. Most of the prior works in this domain leverage variants of Neural Radiance Field (NeRF) [32] as their underlying 3D representations [6, 49], which typically consist of an explicit and structured proxy representation and an implicit feature decoder. However, such hybrid NeRF variants have degraded representation power, particularly when used for generative modeling where a single implicit feature decoder is shared across all objects. Furthermore, the high computational complexity of volumetric rendering leads to both slow rendering speed and extensive memory costs. Recently, the emergence of 3D Gaussian Splatting (GS) [28] has enabled high-quality reconstruction [61, 31, 55] along with real-time rendering speed. The fully explicit characteristic of 3DGS also eliminates the need for a shared implicit decoder. Although 3DGS has been widely studied in scene reconstruction tasks, its spatially unstructured nature presents significant challenge when applying it to generative modeling.

In this work, we introduce GaussianCube, a novel representation crafted to address the unstructured nature of 3DGS and unleash its potential for 3D generative modeling (see Table 1 for comparisons with prior works). Converting 3D Gaussians into a structured format without sacrificing their expressiveness is not a trivial task. We propose to first perform high-quality fitting using a fixed number of Gaussians and then organize them in a spatially structured manner. To keep the number of Gaussians fixed during fitting, a naive solution might omit the densification and pruning steps in GS, which, however, would significantly degrade the fitting quality. In contrast, we propose a *densification-constrained fitting* strategy, which retains the original pruning process yet constrains

[*] Interns at Microsoft Research Asia.

Preprint. Work in progress.

Representation	Spatially-structured	Fully-explicit	High-quality Reconstruction	Efficient Rendering
Vanilla NeRF [32]	✗	✗	✗	✗
Neural Voxels [49]	✓	✗	✗	✗
Triplane [6]	✓	✗	✗	✗
Gaussian Splatting [28]	✗	✓	✓	✓
Our GaussianCube	✓	✓	✓	✓

Table 1: Comparison with prior 3D representations.

the number of Gaussians that perform densification, ensuring the total does not exceed a predefined maximum N^3 (32,768 in this paper). For the subsequent structuralization, we allocate the Gaussians across an $N \times N \times N$ voxel grid using *Optimal Transport (OT)*. Consequently, our fitted Gaussians are systematically arranged within the voxel grid, with each grid containing a Gaussian feature. The proposed OT-based structuralization process achieves maximal spatial coherence, characterized by minimal total transport distances, while preserving the high expressiveness of the 3DGS.

We perform 3D generative modeling with the proposed GaussianCube using diffusion models [23]. The spatially coherent structure of the Gaussians in our representation facilitates efficient feature extraction and permits the use of standard 3D convolutions to capture the correlations among neighboring Gaussians effectively. Therefore, we construct our diffusion model with standard 3D U-Net architecture without elaborate designs. It is worth noting that our diffusion model and the GaussianCube representation are generic, which facilitates both unconditional and conditional generation tasks.

We conduct comprehensive experiments to verify the efficacy of our proposed approach. The model's capability for unconditional generation is evaluated on the ShapeNet dataset [7]. Both the quantitative and qualitative comparisons indicate that our model surpasses all previous methods. Additionally, we perform class-conditioned generation on the OmniObject3D dataset [56], which is a extensive collection of real-world scanned objects with a broad vocabulary. Our model excels in producing semantically accurate 3D objects with complex geometries and realistic textures, outperforming the state-of-the-art methods. These experiments collectively demonstrate the strong capabilities of our GaussianCube and suggest its potential as a powerful and versatile 3D representation for a variety of applications. Some generated samples of our method is presented in Figure 1.

2 Related Work

Radiance field representation. Radiance fields model ray interactions with scene surfaces and can be in either implicit or explicit forms. Early works of neural radiance fields (NeRFs) [32, 65, 37, 1, 39] are often in an implicit form, which represents scenes without defining geometry. These works optimize a continuous scene representation using volumetric ray-marching that leads to extremely high computational costs. Recent works introduce the use of explicit proxy representation followed by an implicit feature decoder to enable faster rendering. The explicit proxy representations directly represent continuous neural features in a discrete data structure, such as triplane [6, 25], voxel grid [15, 43], hash table [34], or point sets [60]. Recently, the 3D Gaussian Splatting methods [28, 61, 55, 12, 30] utilize 3D Gaussians as their underlying representation and adaptively densify and prune them during fitting, which offers impressive reconstruction quality. The fully explicit representation also provides real-time rendering speed. However, the 3D Gaussians are unstructured representation, and require per-scene optimization to achieve photo-realistic quality. In contrast, our work proposes a structured representation termed GaussianCube for 3D generative tasks.

Image-based 3D reconstruction. Compared to per-scene optimization, image-based 3D reconstruction methods [50, 29, 51, 63] can directly reconstruct 3D assets given images without optimization. PixelNeRF [63] leverages an image feature encoder to empower the generalizability of NeRF. Similarly, pixel-aligned Gaussian approaches [8, 45, 47] follow this idea to design feed-forward Gaussian reconstruction networks. LRM [24, 21] shows that transformers can also be scaled up for 3D reconstruction with large-scale training data, which is followed by hybrid Gaussian-triplane methods [67, 58] within the LRM frameworks. However, the limited number of Gaussians and spatially unstructured property hinders these methods from achieving high-quality reconstruction, which also makes it hard to extend them to 3D generative modeling.

3D generation. Previous works of SDS-based optimization [38, 48, 59, 54, 44, 11, 10] distill 2D diffusion priors [40] to a 3D representation with the score functions. Despite the acceleration [46, 62]

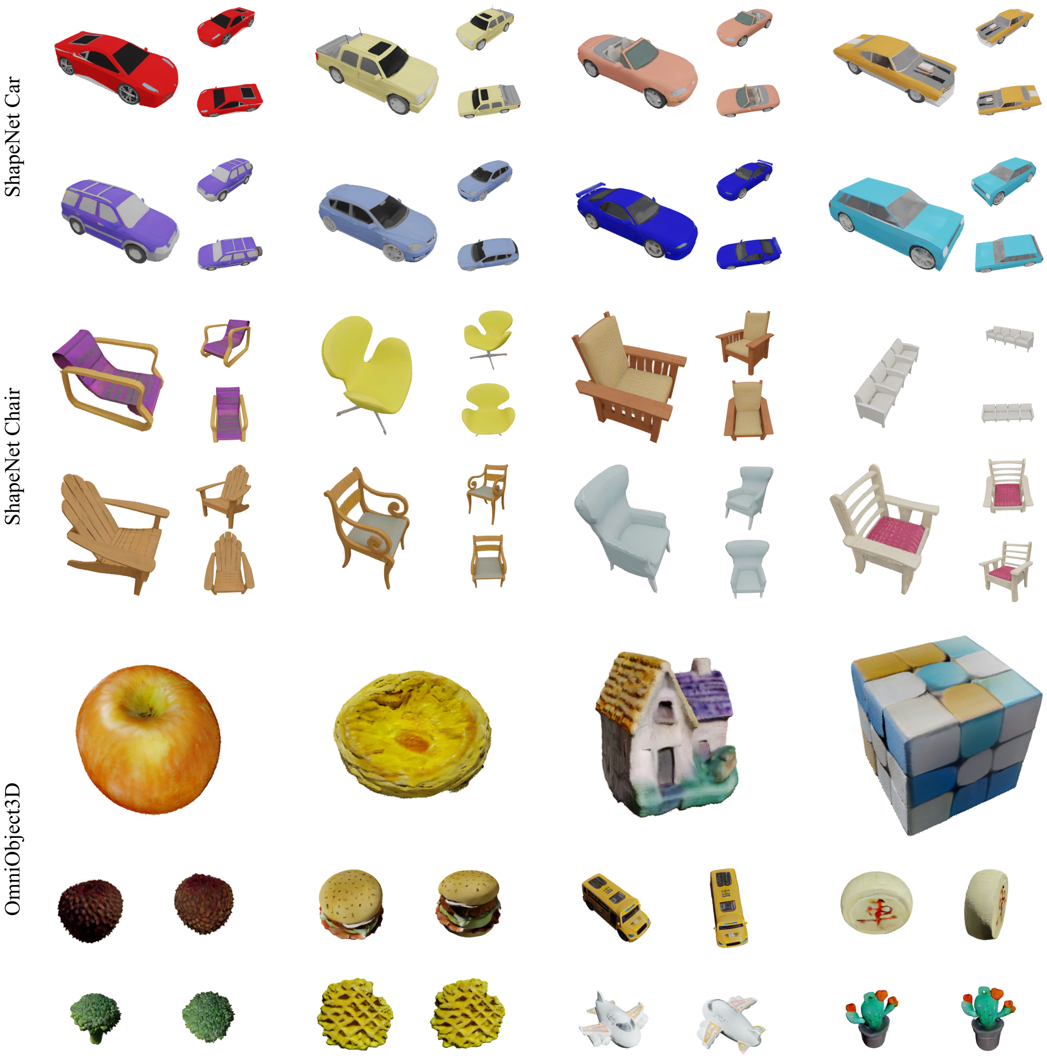


Figure 1: Samples of our generated 3D objects. Our model is able to create diverse objects with complex geometry and rich texture details.

achieved by replacing NeRF with 3D Gaussians, generating high-fidelity 3D Gaussians using these optimization-based methods still requires costly test-time optimization. 3D-aware GANs [6, 16, 5, 18, 36, 13, 57] can generate view-dependent images by training on single image collections. Nevertheless, they fall short in modeling diverse objects with complex geometry variations. Many recent works [53, 33, 19, 49, 41] apply diffusion models for 3D generation using structured proxy 3D representations such as hybrid triplane [53, 41] or voxels [33, 49]. However, they typically need a shared implicit feature decoder across different assets, which greatly limits the representation expressiveness. Also, the inherent computational cost from NeRF leads to slow rendering speed, making it unsuitable for efficient training and rendering. Building upon the strong capability and rendering efficiency of Gaussian Splatting [28], we propose a spatially structured Gaussian representation, making it suitable for 3D generative modeling. A concurrent work of [20] also investigated transforming 3DGS into a volumetric representation. Their method confines the Gaussians to voxel grids during fitting and incorporates a specialized desification strategy. In contrast, our method only restricts the total number of Gaussians, adhering to the original splitting strategy and allowing unrestricted spatial distribution. This preserves the representation power during fitting. The subsequent OT-based voxelization yields spatially coherent arrangement with minimal global offset cost and hence effectively eases the difficulty of generative modeling.

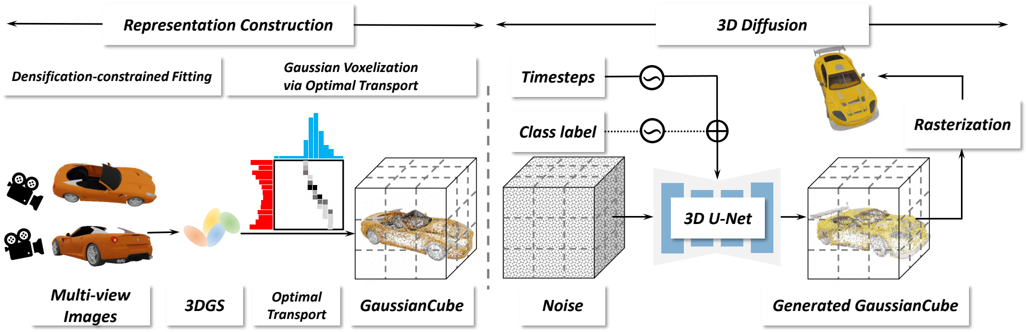


Figure 2: **Overall framework.** Our framework comprises two main stages of representation construction and 3D diffusion. In the representation construction stage, given multi-view renderings of a 3D asset, we perform *densification-constrained fitting* to obtain 3D Gaussians with constant numbers. Subsequently, the Gaussians are voxelized into GaussianCube via *Optimal Transport*. In the 3D diffusion stage, our *3D diffusion model* is trained to generate GaussianCube from Gaussian noise.

3 Method

Following prior works, our framework comprises two primary stages: representation construction and diffusion modeling. In representation construction phase, we first apply a densification-constrained 3DGS fitting algorithm for each object to obtain a constant number of Gaussians. These Gaussians are then organized into a spatially structured representation via Optimal Transport between the positions of Gaussians and centers of a predefined voxel grid. For diffusion modeling, we train a 3D diffusion model to learn the distribution of GaussianCubes. The overall framework is illustrated in Figure 2. We will detail our designs for each stage subsequently.

3.1 Representation Construction

We expect the 3D representation to be both structured, expressive and efficient. Despite Gaussian Splatting (GS) offers superior expressiveness and efficiency against NeRFs, it fails to yield fixed-length representations across different 3D assets; nor does it organize the data in a spatially structured format. To address these limitations, we introduce GaussianCube, which effectively overcomes the unstructured nature of Gaussian Splatting, while retaining both expressiveness and efficiency.

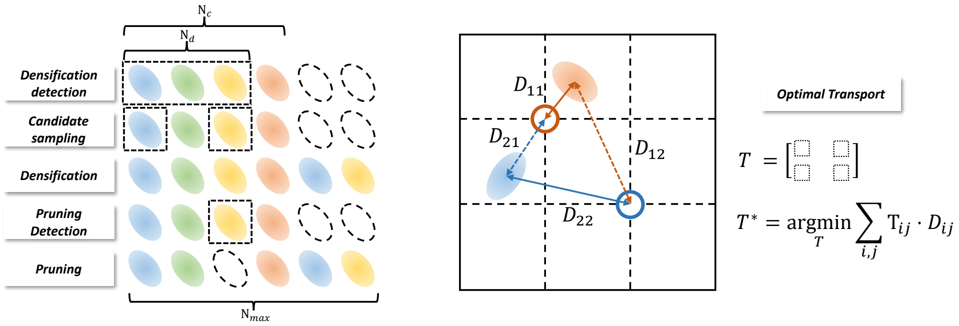
Formally, a 3D asset is represented by a collection of 3D Gaussians as introduced in Gaussian Splatting [28]. The geometry of the i-th 3D Gaussian $\boldsymbol{g}_i$ is given by

$$\boldsymbol{g}_i(\boldsymbol{x}) = \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\right), \quad (1)$$

where $\boldsymbol{\mu}_i \in \mathbb{R}^3$ is the center of the Gaussian and $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3\times 3}$ is the covariance matrix defining the shape and size, which can be decomposed into a quaternion $\boldsymbol{q}_i \in \mathbb{R}^4$ and a vector $\boldsymbol{s}_i \in \mathbb{R}^3$ for rotation and scaling, respectively. Moreover, each Gaussian $\boldsymbol{g}_i$ have an opacity value $\alpha_i \in \mathbb{R}$ and a color feature $\boldsymbol{c}_i \in \mathbb{R}^3$ for rendering. Combining them together, the C-channel feature vector $\boldsymbol{\theta}_i = \{\boldsymbol{\mu}_i, \boldsymbol{s}_i, \boldsymbol{q}_i, \alpha_i, \boldsymbol{c}_i\} \in \mathbb{R}^C$ fully characterizes the Gaussian $\boldsymbol{g}_i$.

Notably, the adaptive control is one of the most essential steps during the fitting process in GS [28]. It dynamically clones Gaussians in under-reconstructed regions, splits Gaussians in over-reconstructed regions, and eliminates those with irregular dimensions. Although the adaptive control substantially improves the fitting quality, it can lead to a varying number of Gaussians for different objects. Furthermore, the Gaussians are stored without a predetermined spatial order, resulting in an absence of an organized spatial structure. These aspects pose significant challenges to 3D generative modeling. To overcome these obstacles, we first introduce our densification-constrained fitting strategy to obtain a fixed number of free Gaussians. Then, we systematically arrange the resulting Gaussians within a predefined voxel grid via Optimal Transport, thereby achieving a spatially structured GS representation.

Densification-constrained fitting. Our approach begins with the aim of maintaining a constant number of Gaussians $\boldsymbol{g} \in \mathbb{R}^{N_{\max}\times C}$ across different objects during the fitting. A naive approach



(a) (b)

Figure 3: **Illustration of representation construction.** First, we perform densification-constrained fitting to yield a fixed number of Gaussians, as shown in (a). We then employ Optimal Transport to organize the resultant Gaussians into a voxel grid. A 2D illustration of this process is presented in (b).

might involve omitting the densification and pruning steps in the original GS. However, we argue that such simplifications significantly harm the fitting quality, with empirical evidence shown in Table 4. Instead, we propose to retain the pruning process while imposing a new constraint on the densification phase. Specifically, if the current iteration comprises N_{c} Gaussians and N_d Gaussians need to be densified, we introduce a measure to prevent exceeding the predefined maximum of N_{max} Gaussians (with N_{max} set to $32,768$ in this work). This is achieved by selecting $N_{\text{max}} - N_{\text{c}}$ Gaussians with the largest view-space positional gradients from the N_d candidates for densification in cases where $N_d > N_{\text{max}} - N_{\text{c}}$. Otherwise, all N_d Gaussians are subjected to densification as in the original GS. Additionally, instead of performing the cloning and splitting in the same densification steps, we opt to perform each alternately without influencing each other. Upon completion of the entire fitting process, we pad Gaussians with $\alpha = 0$ to reach the target count of N_{max} without affecting the rendering results. The detailed fitting procedure is shown in Figure 3 (a).

Gaussian voxelization via Optimal Transport. To further organize the obtained Gaussians into a spatially structured representation for 3D generative modeling, we propose to map the Gaussians to a predefined structured voxel grid $\boldsymbol{v} \in \mathbb{R}^{N_v \times N_v \times N_v \times C}$ where $N_v = \sqrt[3]{N_{\text{max}}}$. Intuitively, we aim to "move" each Gaussian into a voxel grid while preserving their geometric relations as much as possible. To this end, we formulate this as an Optimal Transport (OT) problem [52, 3] between the Gaussians' spatial positions $\{\boldsymbol{\mu}_i, i = 1, \ldots, N_{\text{max}}\}$ and the voxel grid centers $\{\boldsymbol{x}_j, j = 1, \ldots, N_{\text{max}}\}$. Let $\mathbf{D}$ be a distance matrix with $\mathbf{D}_{ij}$ being the moving distance between $\boldsymbol{\mu}_i$ and $\boldsymbol{x}_j$, i.e., $\mathbf{D}_{ij} = \|\boldsymbol{\mu}_i - \boldsymbol{x}_j\|^2$. The transport plan is represented by a binary matrix $\mathbf{T} \in \mathbb{R}^{N_{\text{max}} \times N_{\text{max}}}$, and the optimal transport plan is given by:

$$
\begin{aligned}
\underset{\mathbf{T}}{\text{minimize}} \quad & \textstyle\sum_{i=1}^{N_{\text{max}}} \sum_{j=1}^{N_{\text{max}}} \mathbf{T}_{ij} \mathbf{D}_{ij} \\
\text{subject to} \quad & \textstyle\sum_{j=1}^{N_{\text{max}}} \mathbf{T}_{ij} = N_{\text{max}} \quad \forall i \in \{1, \ldots, N_{\text{max}}\} \\
& \textstyle\sum_{i=1}^{N_{\text{max}}} \mathbf{T}_{ij} = N_{\text{max}} \quad \forall j \in \{1, \ldots, N_{\text{max}}\} \\
& \mathbf{T}_{ij} \in \{0, 1\} \quad \forall (i, j) \in \{1, \ldots, N_{\text{max}}\} \times \{1, \ldots, N_{\text{max}}\}.
\end{aligned}
\tag{2}
$$

The solution is a bijective transport plan $\mathbf{T}^*$ that minimizes the total transport distances. We employ the Jonker-Volgenant algorithm [26] to solve the OT problem. We organize the Gaussians according to the solutions, with the j-th voxel grid encapsulating the feature vector of the corresponding Gaussian $\boldsymbol{\theta}_k = \{\boldsymbol{\mu}_k - \boldsymbol{x}_j, \boldsymbol{s}_k, \boldsymbol{q}_k, \alpha_k, \boldsymbol{c}_k\} \in \mathbb{R}^C$, where k is determined by the optimal transport plan (*i.e.*, $\mathbf{T}^*_{kj} = 1$). Note that we substitute the original Gaussian positions with offsets of the current voxel center to reduce the solution space for diffusion modeling. As a result, our fitted Gaussians are systematically arranged within a voxel grid $\boldsymbol{v}$ and maintain the spatial coherence.

3.2 3D Diffusion on GaussianCube

We now introduce our 3D diffusion model incorporated with the proposed expressive, efficient and spatially structured representation. After organizing the fitted Gaussians $\boldsymbol{g}$ into GaussianCube $\boldsymbol{y}$ for each object, we aim to model the distribution of GaussianCube, *i.e.*, $p(\boldsymbol{y})$.

Formally, the generation procedure can be formulated into the inversion of a discrete-time Markov forward process. During the forward phase, we gradually add noise to $\boldsymbol{y}_0 \sim p(\boldsymbol{y})$ and obtain a sequence of increasingly noisy samples $\{\boldsymbol{y}_t | t \in [0, T]\}$ according to

$$\boldsymbol{y}_t := \alpha_t \boldsymbol{y}_0 + \sigma_t \boldsymbol{\epsilon}, \tag{3}$$

where $\boldsymbol{\epsilon} \in \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ represents the added Gaussian noise, and α_t, σ_t constitute the noise schedule which determines the level of noise added to destruct the original data sample. As a result, $\boldsymbol{y}_T$ will finally reach isotropic Guassian noise after sufficient destruction steps. By reversing the above process, we are able to perform the generation process by gradually denoise the sample starting from pure Gaussian noise $\boldsymbol{y}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ until reaching $\boldsymbol{y}_0$. Our diffusion model is trained to denoise $\boldsymbol{y}_t$ into $\boldsymbol{y}_0$ for each timestep t, facilitating both unconditional and class-conditioned generation.

Model architecture. Thanks to the spatially structured organization of the proposed GaussianCube, standard 3D convolution is sufficient to effectively extract and aggregate the features of neighboring Gaussians without elaborate designs. We leverage the popular U-Net network for diffusion [35, 14] and simply replace the original 2D convolution layer to their 3D counterparts. The upsampling, downsampling and attention operations are also replaced with corresponding 3D implementations.

Conditioning mechanism. When performing class-conditional diffusion training, we use adaptive group normalization (AdaGN) [14] to inject conditions of class labels $\boldsymbol{c}_{\text{cls}}$ into our model, which can be defined as:

$$\operatorname{AdaGN}(\boldsymbol{f}_i) = \operatorname{GroupNorm}(\boldsymbol{f}_i) \cdot (1 + \boldsymbol{\gamma}) + \boldsymbol{\beta}, \tag{4}$$

where the group-wise scale and shift parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are estimated to modulate the activations $\{\boldsymbol{f}_i\}$ in each residual block from the embeddings of both timesteps t and condition $\boldsymbol{c}_{\text{cls}}$.

Training objective. In our 3D diffusion training, we parameterize our model $\hat{\boldsymbol{y}}_\theta$ to predict the noise-free input $\boldsymbol{y}_0$ using:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{t, \boldsymbol{y}_0, \boldsymbol{\epsilon}} \left[\left\| \hat{\boldsymbol{y}}_\theta \left(\alpha_t \boldsymbol{y}_0 + \sigma_t \boldsymbol{\epsilon}, t, \boldsymbol{c}_{\text{cls}} \right) - \boldsymbol{y}_0 \right\|_2^2 \right], \tag{5}$$

where the condition signal $\boldsymbol{c}_{\text{cls}}$ is only needed when training class-conditioned diffusion models. We additionally add supervision on the image level to ensure better rendering quality of generated GaussianCube, which has been shown to effectively enhance the visual quality in previous works [53, 33]. Specifically, we penalize the discrepancy between the rasterized images I_{pred} of the predicted GaussianCubes and the ground-truth images I_{gt}:

$$\begin{aligned} \mathcal{L}_{\text{image}} &= \mathcal{L}_{\text{pixel}} + \mathcal{L}_{\text{perc}} \\ &= \mathbb{E}_{t, I_{\text{pred}}} \left(\sum_l \left\| \Psi^l \left(I_{\text{pred}} \right) - \Psi^l \left(I_{\text{gt}} \right) \right\|_2^2 \right) \\ &+ \mathbb{E}_{t, I_{\text{pred}}} \left(\left\| I_{\text{pred}} - I_{\text{gt}} \right\|_2^2 \right), \end{aligned} \tag{6}$$

where Ψ^l is the multi-resolution feature extracted using the pre-trained VGG [42]. Benefiting from the efficiency of both rendering speed and memory costs from Gaussian Splatting [28], we are able to render the full image rather than only a small patch as in previous NeRF-based methods [53, 9], which facilitates fast training with high-resolution renderings. Our overall training loss can be formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{simple}} + \lambda \mathcal{L}_{\text{image}}, \tag{7}$$

where λ is a balancing weight.

4 Experiments

4.1 Dataset and Metrics

To measure the expressiveness and efficiency of various 3D representations, we fit 100 objects in ShapeNet Car [7] using each representation and report the PSNR, LPIPS [66] and Structural Similarity Index Measure (SSIM) metrics when synthesizing novel views. Furthermore, we conduct experiments of single-category unconditional generation on ShapeNet [7] Car and Chair. We randomly render 150 views and fit $32 \times 32 \times 32 \times 14$ GaussianCube for each object. To further validate the strong

Representation	Spatially-structured	PSNR↑	LPIPS↓	SSIM↑	Rel. Speed↑	Params (M)↓
Instant-NGP	✗	33.98	0.0386	0.9809	1×	12.25
Gaussian Splatting	✗	**35.32**	**0.0303**	**0.9874**	2.58×	1.84
Voxels	✓	28.95	0.0959	0.9470	1.73×	0.47
Voxels*	✓	25.80	0.1407	0.9111	1.73×	0.47
Triplane	✓	32.61	0.0611	0.9709	1.05×	6.30
Triplane*	✓	31.39	0.0759	0.9635	1.05×	6.30
Our GaussianCube	✓	34.94	0.0347	0.9863	**3.33×**	**0.46**

Table 2: Quantitative results of representation fitting on ShapeNet Car. * denotes that the implicit feature decoder is shared across different objects.

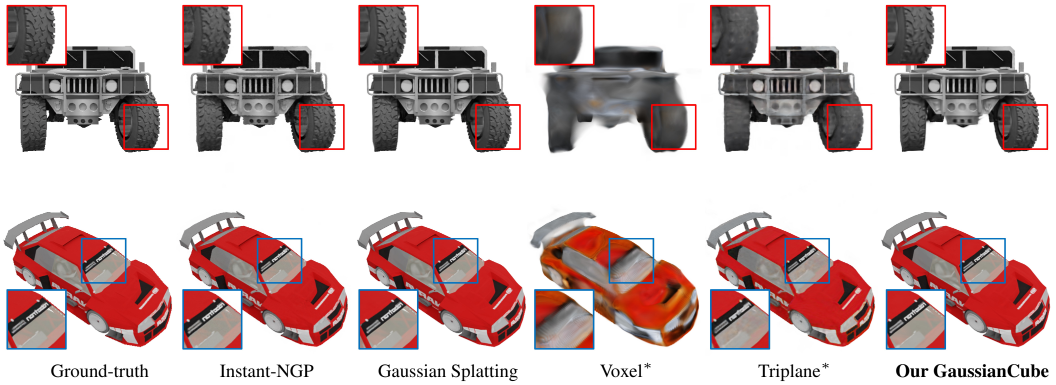

Ground-truth Instant-NGP Gaussian Splatting Voxel* Triplane* **Our GaussianCube**

Figure 4: Qualitative results of object fitting.

capability of the proposed framework, we also conduct experiments on Omniobject3D [56], which is a challenging dataset containing large-vocabulary real-world scanned 3D objects. We fit GaussianCube of the same dimensions as ShapeNet using 100 multi-view renderings for each object. To numerically measure the generation quality, we report the FID [22] and KID [2] scores between 50K renderings of generated samples and 50K ground-truth renderings at the 512×512 resolution.

4.2 Implementation Details

To construct GaussianCube for each object, we perform the proposed densification-constrained fitting for 30K iterations. Since the time complexity of Jonker-Volgenant algorithm [26] is $O\left(N_{\text{max}}^3\right)$, we opt for an approximate solution to the Optimal Transport problem. This is achieved by dividing the positions of the Gaussians and the voxel grid into four sorted segments and then applying the Jonker-Volgenant solver to each segment individually. We empirically found this approximation successfully strikes a balance between computational efficiency and spatial structure preservation. For the 3D diffusion model, we adopt the ADM U-Net network [35, 14]. We perform full attention at the resolution of 8^3 and 4^3 within the network. The timesteps of diffusion models are set to $1,000$ and we train the models using the cosine noise schedule [35] with loss weight λ set to 10. All models are trained on 16 Tesla V100 GPUs with a total batch size of 128.

4.3 Main Results

3D fitting. We first evaluate the representation power of our GaussianCube using 3D object fitting and compare it against previous NeRF-based representations including Triplane and Voxels, which are widely adopted in 3D generative modeling [6, 53, 4, 33, 49]. We also include Instant-NGP [34] and original Gaussian Splatting [28] for reference, although they cannot be directly applied in generative modeling due to their spatially unstructured nature. As shown in Table 2, our GaussianCube outperforms all NeRF-based representations among all metrics. The visualizations in Figure 3 illustrate that GaussianCube can faithfully reconstruct geometry details and intricate textures, which demonstrates its strong capability. Moreover, compared with over $131,000$ Gaussians utilized in the original GS for each object, our GaussianCube only employs 4× less Gaussians due to our densification-constrained fitting. This leads to the faster fitting speed of our method and significantly fewer parameters (over 10× less than Triplane), which demonstrate its efficiency and compactness.

Method	ShapeNet Car		ShapeNet Chair		OmniObject3D	
	FID-50K↓	KID-50K(‰)↓	FID-50K↓	KID-50K(‰)↓	FID-50K↓	KID-50K(‰)↓
EG3D	30.48	20.42	27.98	16.01	-	-
GET3D	17.15	9.58	19.24	10.95	-	-
DiffTF	51.88	41.10	47.08	31.29	46.06	22.86
Ours	**13.01**	**8.46**	**15.99**	**9.95**	**11.62**	**2.78**

Table 3: Quantitative results of unconditional generation on ShapeNet Car and Chair [7] and class-conditioned generation on OmniObject3D [56].

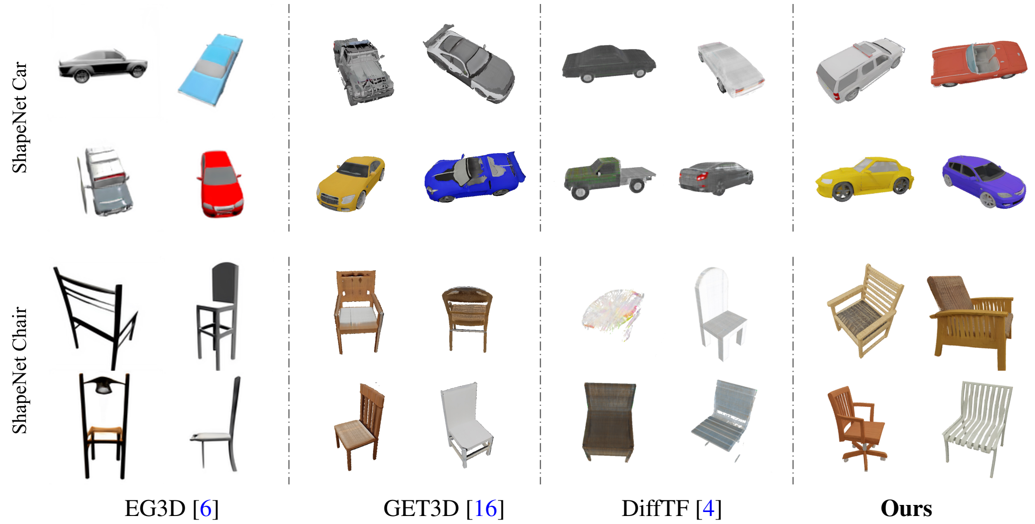


Figure 5: Qualitative comparison of unconditional 3D generation on ShapeNet Car and Chair datasets. Our model is capable of generating results of complex geometry with rich details.

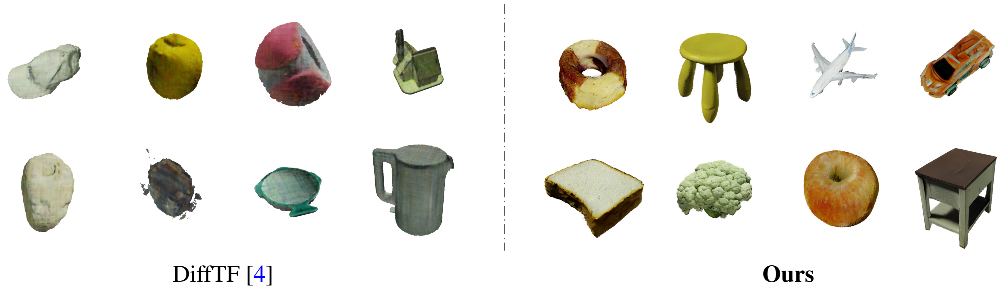


Figure 6: Qualitative comparison of class-conditioned 3D generation on large-vocabulary OmniObject3D [56]. Our model is able to handle diverse distribution with high-fidelity generated samples.

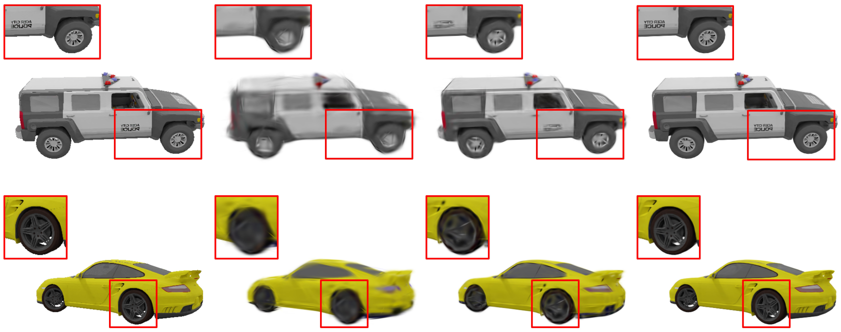
Notably, for 3D generation tasks, NeRF-based methods typically necessitate a shared implicit feature decoder for different objects, which leads to significant decreases sin fitting quality compared to single-object fitting, as shown in Table 2. In contrast, the explicit characteristic of GS allows our GaussianCube to bypass such a shared feature decoder, resulting in no quality gap between single and multiple object fitting.

Single-category unconditional generation. For unconditional generation, we compare our method with the state-of-the-art 3D generation works including 3D-aware GANs [6, 16] and Triplane diffusion models [4]. As shown in Table 3, our method surpasses all prior works in terms of both FID and KID scores and sets new records. We also provide visual comparisons in Figure 5, where EG3D and DiffTF tend to generate blurry results with poor geometry, and GET3D fails to provide satisfactory textures. In contrast, our method yields high-fidelity results with authentic geometry and sharp texture details.

Large-vocabulary class-conditioned generation. We also compare class-conditioned generation with DiffTF [4] on more diverse and challenging OmniObject3D [56] dataset. We achieve significantly

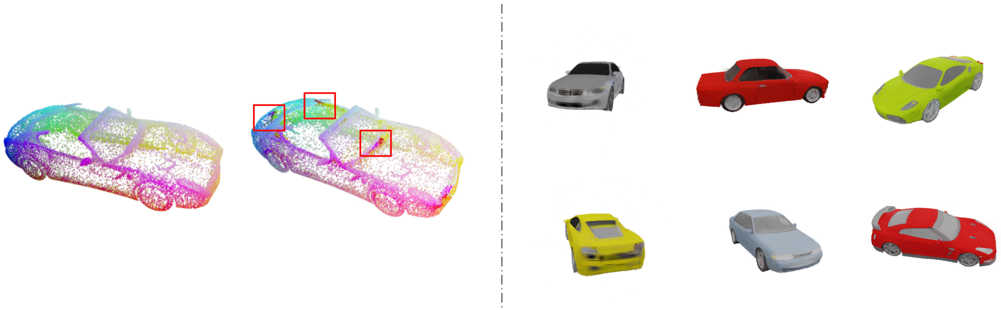
Method	**Densify & Prune**	**Representation Fitting**			**Generation**	
		PSNR↑	**LPIPS↓**	**SSIM↑**	**FID-50K↓**	**KID-50K(‰)↓**
A. Voxel grid w/o offset	✗	25.87	0.1228	0.9217	-	-
B. Voxel grid w/ offset	✗	30.18	0.0780	0.9628	40.52	24.35
C. Ours w/o OT	✓	**34.94**	**0.0346**	**0.9863**	21.41	14.37
D. Ours	✓	**34.94**	**0.0346**	**0.9863**	**13.01**	**8.46**

Table 4: Quantitative ablation of both representation fitting and generation quality on ShapeNet Car.



Ground-truth Table 4 A. Table 4 B. **Table 4 D. (Ours)**

Figure 7: Qualitative ablation of representation fitting. Our GaussianCube achieves superior fitting results while maintaining a spatial structure.



Optimal Transport (Ours) Nearest Neighbor Transport

(a)

Table 4 B. Table 4 C. **Table 4 D. (Ours)**

(b)

Figure 8: Ablation study of the Gaussian structuralization methods and 3D generation. For visualization of Gaussian structuralization in (a), we map the coordinates of the corresponding voxel grid of each Gaussians to RGB values to visualize the organization. Our Optimal Transport solution yields smooth transit among Gaussians, indicating a coherent global structure, whereas Nearest Neighbor Transport leads to obvious discontinuities. Our OT-based solution also results in the best generation quality shown in (b).

better FID and KID scores than DiffTF as shown in Table 3, demonstrating the stronger capacity of our method. Visual comparisons in Figure 6 reveal that DiffTF often struggles to create intricate geometry and detailed textures, whereas our method is able to generate objects with complex geometry and realistic textures.

4.4 Ablation Study

We first examine the key factors in representation construction. To spatially structure the Gaussians, a simplest approach would be anchoring the positions of Gaussians to a predefined voxel grid while omitting densification and pruning, which leads to severe failure when fitting the objects as shown in Figure 7. Even by introducing learnable offsets to the voxel grid, the complex geometry and detailed textures still can not be well captured, as shown in Figure 7. We observe that the offsets are typically too small to effectively lead the Gaussians close to the object surfaces, which demonstrates the necessity of performing densification and pruning during object fitting. Based on these insights, we do not organize the Gaussians during the fitting stage. Instead, we only maintain a constant number of Gaussians using densification-constrained fitting and post-process the Gaussians into a spatially structured representation. Our GaussianCube can capture both complex geometry and intricate details as illustrated in Figure 7. The numerical comparison in Table 4 also demonstrates the superior fitting quality of our GaussianCube.

We also evaluate how the representation affects 3D generative modeling in Table 4 and Figure 8. Limited by the poor fitting quality, performing diffusion modeling on voxel grid with learnable offsets leads to blurry generation results as shown in Figure 8. To validate the importance of organizing Gaussians via OT, we compare with the organization based on nearest neighbor transport between positions of Gaussians and centers of voxel grid. We linearly map each Gaussian's corresponding coordinates of voxel grid to RGB color to visualize the different organizations. As shown in Figure 8 (a), our proposed OT approach results in smooth color transitions, indicating that our method successfully preserves the spatial correspondence. However, nearest neighbor transport does not consider global structure, which leads to abrupt color transitions. Notably, since our OT-based organization considers the global spatial coherence, both the quantitative results in Table 4 and visual comparisons Figure 8 indicate that our structured arrangement facilitates generative modeling by alleviating its complexity, successfully leading to superior generation quality.

5 Conclusion

We have presented GaussianCube, a novel representation crafted for 3D generative models. We address the unstructured nature of Gaussian Splatting and unleash its potential for 3D generative modeling. Firstly, we fit each 3D object with a constant number of Gaussians by our proposed densification-constrained fitting algorithm. Furthermore, we organize the obtained Gaussians into a spatially structured representation by solving the Optimal Transport problem between the positions of Gaussians and the predefined voxel grid. The proposed GaussianCube is expressive, efficient and with spatially coherent structure, providing a strong 3D representation alternative for 3D generation. We train 3D diffusion models to perform generative modeling using GaussianCube, and achieve state-of-the-art generation quality on the evaluated datasets without elaborate network or training algorithm design. This demonstrates the promise of GaussianCube to be a versatile and powerful 3D representation for 3D generation.

References

[1] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022.

[2] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018.

[3] Rainer E Burkard and Eranda Cela. Linear assignment problems and extensions. In *Handbook of combinatorial optimization: Supplement volume A*, pages 75–149. Springer, 1999.

[4] Ziang Cao, Fangzhou Hong, Tong Wu, Liang Pan, and Ziwei Liu. Large-vocabulary 3d diffusion model with transformer. *arXiv preprint arXiv:2309.07920*, 2023.

[5] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5799–5809, 2021.

[6] 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, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16123–16133, 2022.

[7] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[8] David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. *arXiv preprint arXiv:2312.12337*, 2023.

[9] Xingyu Chen, Yu Deng, and Baoyuan Wang. Mimic3d: Thriving 3d-aware gans via 3d-to-2d imitation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2338–2348. IEEE Computer Society, 2023.

[10] Yongwei Chen, Tengfei Wang, Tong Wu, Xingang Pan, Kui Jia, and Ziwei Liu. Comboverse: Compositional 3d assets creation using spatially-aware diffusion guidance. *arXiv preprint arXiv:2403.12409*, 2024.

[11] Yiji Cheng, Fei Yin, Xiaoke Huang, Xintong Yu, Jiaxiang Liu, Shikun Feng, Yujiu Yang, and Yansong Tang. Efficient text-guided 3d-aware portrait generation with score distillation sampling on distribution. *arXiv preprint arXiv:2306.02083*, 2023.

[12] R James Cotton and Colleen Peyton. Dynamic gaussian splatting from markerless motion capture reconstruct infants movements. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 60–68, 2024.

[13] Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. Gram: Generative radiance manifolds for 3d-aware image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10673–10683, 2022.

[14] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

[15] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022.

[16] 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. *arXiv preprint arXiv:2209.11163*, 2022.

[17] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[18] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021.

[19] Anchit Gupta, Wenhan Xiong, Yixin Nie, Ian Jones, and Barlas Oğuz. 3dgen: Triplane latent diffusion for textured mesh generation. *arXiv preprint arXiv:2303.05371*, 2023.

[20] Xianglong He, Junyi Chen, Sida Peng, Di Huang, Yangguang Li, Xiaoshui Huang, Chun Yuan, Wanli Ouyang, and Tong He. Gvgen: Text-to-3d generation with volumetric representation. *arXiv preprint arXiv:2403.12957*, 2024.

[21] Zexin He and Tengfei Wang. Openlrm: Open-source large reconstruction models, 2023.

[22] 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. *Advances in neural information processing systems*, 30, 2017.

[23] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[24] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023.

[25] Wenbo Hu, Yuling Wang, Lin Ma, Bangbang Yang, Lin Gao, Xiao Liu, and Yuewen Ma. Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19774–19783, 2023.

[26] Roy Jonker and Ton Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. In *DGOR/NSOR: Papers of the 16th Annual Meeting of DGOR in Cooperation with NSOR/Vorträge der 16. Jahrestagung der DGOR zusammen mit der NSOR*, pages 622–622. Springer, 1988.

[27] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[28] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023.

[29] Hao Li, Bart Adams, Leonidas J Guibas, and Mark Pauly. Robust single-view geometry and motion reconstruction. *ACM Transactions on Graphics (ToG)*, 28(5):1–10, 2009.

[30] Mengtian Li, Shengxiang Yao, Zhifeng Xie, Keyu Chen, and Yu-Gang Jiang. Gaussianbody: Clothed human reconstruction via 3d gaussian splatting. *arXiv preprint arXiv:2401.09720*, 2024.

[31] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023.

[32] 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. *Communications of the ACM*, 65(1):99–106, 2021.

[33] Norman Müller, Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulo, Peter Kontschieder, and Matthias Nießner. Diffrf: Rendering-guided 3d radiance field diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4328–4338, 2023.

[34] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4): 1–15, 2022.

[35] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[36] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021.

[37] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021.

[38] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[39] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021.

[40] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[41] J Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3d neural field generation using triplane diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20875–20886, 2023.

[42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[43] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022.

[44] Jingxiang Sun, Bo Zhang, Ruizhi Shao, Lizhen Wang, Wen Liu, Zhenda Xie, and Yebin Liu. Dreamcraft3d: Hierarchical 3d generation with bootstrapped diffusion prior. *arXiv preprint arXiv:2310.16818*, 2023.

[45] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. *arXiv preprint arXiv:2312.13150*, 2023.

[46] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*, 2023.

[47] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. *arXiv preprint arXiv:2402.05054*, 2024.

[48] Junshu Tang, Tengfei Wang, Bo Zhang, Ting Zhang, Ran Yi, Lizhuang Ma, and Dong Chen. Make-it-3d: High-fidelity 3d creation from a single image with diffusion prior. *arXiv preprint arXiv:2303.14184*, 2023.

[49] Zhicong Tang, Shuyang Gu, Chunyu Wang, Ting Zhang, Jianmin Bao, Dong Chen, and Baining Guo. Volumediffusion: Flexible text-to-3d generation with efficient volumetric encoder. *arXiv preprint arXiv:2312.11459*, 2023.

[50] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3405–3414, 2019.

[51] Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2626–2634, 2017.

[52] Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

[53] Tengfei Wang, Bo Zhang, Ting Zhang, Shuyang Gu, Jianmin Bao, Tadas Baltrusaitis, Jingjing Shen, Dong Chen, Fang Wen, Qifeng Chen, et al. Rodin: A generative model for sculpting 3d digital avatars using diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4563–4573, 2023.

[54] Zhenwei Wang, Tengfei Wang, Gerhard Hancke, Ziwei Liu, and Rynson W. H. Lau. Themestation: Generating theme-aware 3d assets from few exemplars. *ArXiv*, 2024.

[55] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023.

[56] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, et al. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 803–814, 2023.

[57] Jianfeng Xiang, Jiaolong Yang, Yu Deng, and Xin Tong. Gram-hd: 3d-consistent image generation at high resolution with generative radiance manifolds. *arXiv preprint arXiv:2206.07255*, 2022.

[58] Dejia Xu, Ye Yuan, Morteza Mardani, Sifei Liu, Jiaming Song, Zhangyang Wang, and Arash Vahdat. Agg: Amortized generative 3d gaussians for single image to 3d. *arXiv preprint arXiv:2401.04099*, 2024.

[59] Jiale Xu, Xintao Wang, Weihao Cheng, Yan-Pei Cao, Ying Shan, Xiaohu Qie, and Shenghua Gao. Dream3d: Zero-shot text-to-3d synthesis using 3d shape prior and text-to-image diffusion models. *arXiv preprint arXiv:2212.14704*, 2022.

[60] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5438–5448, 2022.

[61] Yuelang Xu, Benwang Chen, Zhe Li, Hongwen Zhang, Lizhen Wang, Zerong Zheng, and Yebin Liu. Gaussian head avatar: Ultra high-fidelity head avatar via dynamic gaussians. *arXiv preprint arXiv:2312.03029*, 2023.

[62] Taoran Yi, Jiemin Fang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. *arXiv preprint arXiv:2310.08529*, 2023.

[63] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021.

[64] Bowen Zhang, Shuyang Gu, Bo Zhang, Jianmin Bao, Dong Chen, Fang Wen, Yong Wang, and Baining Guo. Styleswin: Transformer-based gan for high-resolution image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11304–11314, 2022.

[65] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020.

[66] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[67] Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. *arXiv preprint arXiv:2312.09147*, 2023.