GaussianCube: A Structured and Explicit Radiance Representation 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

We introduce a radiance representation that is both structured and fully explicit and thus greatly facilitates 3D generative modeling. Existing radiance representations either require an implicit feature decoder, which significantly degrades the modeling power of the representation, or are spatially unstructured, making them difficult to integrate with mainstream 3D diffusion methods. We derive GaussianCube by first using a novel densification-constrained Gaussian fitting algorithm, which yields high-accuracy fitting using a fixed number of free Gaussians, and then rearranging these Gaussians into a predefined voxel grid via Optimal Transport. Since GaussianCube is a structured grid representation, it allows us to use standard 3D U-Net as our backbone in diffusion modeling without elaborate designs. More importantly, the high-accuracy fitting of the Gaussians allows us to achieve a high-quality representation with orders of magnitude fewer parameters than previous structured representations for comparable quality, ranging from one to two orders of magnitude. The compactness of GaussianCube greatly eases the difficulty of 3D generative modeling. Extensive experiments conducted on unconditional and class-conditioned object generation, digital avatar creation, and text-to-3D synthesis all show that our model achieves state-of-the-art generation results both qualitatively and quantitatively, underscoring the potential of GaussianCube as a highly accurate and versatile radiance representation for 3D generative modeling. Project page: https://gaussiancube.github.io/.

1 Introduction

The field of 3D generation [59, 39, 5, 57, 49, 8, 19, 11, 61, 76] has witnessed remarkable growth, driven by advancements in generative modeling [25, 20, 41, 17, 72, 29]. Most of the prior works in this domain leverage variants of Neural Radiance Field (NeRF) [38, 8, 57, 40] as their underlying 3D representations, which typically consist of an explicit structured proxy representation and an implicit feature decoder. However, such hybrid NeRF variants exhibit degraded representation power, particularly when used for generative modeling where a single implicit feature decoder is shared across all objects. Additionally, 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) [30] has enabled improved reconstruction quality and real-time rendering capabilities [69, 36, 63, 35]. The fully explicit nature of 3DGS eliminates the need for a shared implicit decoder, providing another key advantage over NeRFs. Although 3DGS has been widely studied in scene reconstruction tasks, its spatially unstructured nature presents a significant challenge when applied to mainstream generative modeling frameworks.

[*]Interns at Microsoft Research Asia. [†]Equal advising. [‡]Corresponding authors.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

https://doi.org/10.52202/079017-3091

Representation	Spatially-structured	Fully-explicit	Real-time Rendering	Rel. Parameters↓
Instant-NGP [38]	✗	✗	✗	26.63×
Neural Voxels [57]	✓	✗	✗	145.9×
Triplane [8]	✓	✗	✗	13.7×
Gaussian Splatting [30]	✗	✓	✓	4.0×
Our GaussianCube	✓	✓	✓	**1.0×**

Table 1: Comparison with previous 3D representations with respect to spatial structure, explicitness, real-time rendering capability, and relative parameter count (Rel. Parameters) for representations of comparable quality.

To overcome these barriers, we introduce GaussianCube – an innovative radiance representation that is both structured and fully explicit, with strong fitting capabilities (see Table 1 for comparisons with prior works). The proposed approach first ensures high-accuracy fitting with a predefined number of free Gaussians, and subsequently organizes these Gaussians into a structured voxel grid. Such an explicit grid-based structure permits the seamless application of standard 3D convolutional architectures, such as U-Net, thereby eliminating the need for complex, specialized network designs [77, 59] that are often necessary with unstructured or implicitly decoded representations.

Structuring 3D Gaussians without sacrificing fitting quality is not a trivial task. A naive starting point would be obtaining a fixed number of Gaussians by omitting the densification and pruning steps in GS. However, such simplification fails to lead the Gaussians close to the object surfaces and results in significant quality degradation. In contrast, we propose a *densification-constrained fitting* strategy, which retains the original pruning process yet constrains the number of Gaussians that perform densification, ensuring the total does not exceed a predefined maximum N^3. 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 voxel containing the features of a Gaussian. The proposed OT-based structuralization achieves maximal spatial correspondence, characterized by minimal total transport distances, while preserving the expressive power of 3DGS.

The structured nature of GaussianCube enables us to perform efficient 3D diffusion [25] modeling for the following three reasons: 1) It allows the use of standard 3D U-Net as our backbone for diffusion modeling without elaborate designs. 2) The spatial coherence of GaussianCube permits the use of standard 3D convolutions to capture the correlations among neighboring Gaussians, facilitating efficient feature extraction. 3) GaussianCube enables high-quality fitting with orders of magnitude fewer parameters than prior structured representations of similar quality. Since recent works [32, 3] have demonstrated diffusion models' struggle in handling high-dimensional distributions, the compactness of GaussianCube significantly reduces the modeling difficulty of the generative framework.

We conduct comprehensive experiments to verify the efficacy of our approach. The model's capability for unconditional and class-conditioned generation is evaluated on the ShapeNet [9] and OmniObject3D [64] datasets. Both the quantitative and qualitative comparisons indicate that our model surpasses all previous methods. We also perform digital avatar generation on a synthetic avatar dataset [62]. Our model is capable of producing high-fidelity 3D avatars conditioned on single portrait images, excelling beyond prior art in both identity preservation and detail creation. Additionally, we assess our model's capacity for the challenging text-to-3D creation task on Objaverse [14]. Our model demonstrates competitive performance both quantitatively and qualitatively, producing results consistent with the given text prompts in just 2.3 seconds. All experiments show 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 are 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) [38, 74, 43, 1, 45] 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 [8, 26, 18, 51, 40, 68] followed by an implicit feature decoder to enable faster rendering. Recently, the 3D Gaussian Splatting methods [30, 69, 63, 13, 31, 10, 71] utilize 3D Gaussians as their underlying representation and offer impressive reconstruction quality. The fully explicit representation also provides real-time rendering speed. However, the 3D Gaussians are unstructured representation, and

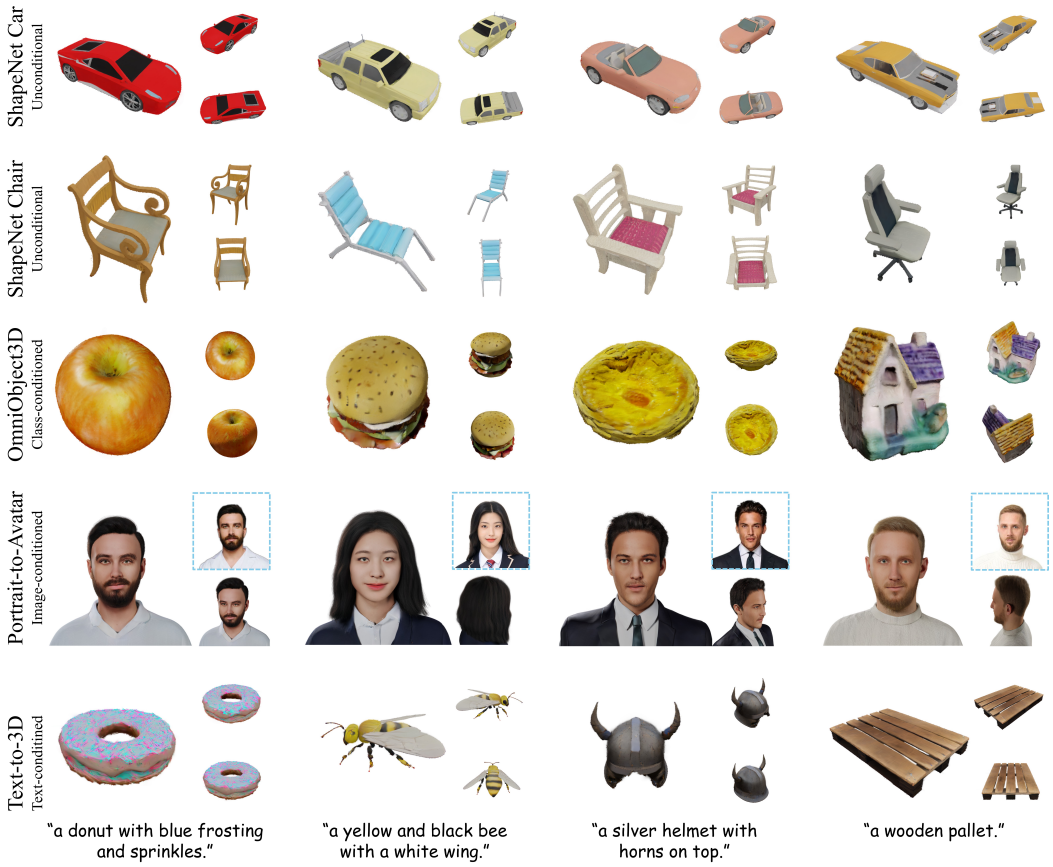


Figure 1: Our diffusion model is able to create diverse objects with complex geometry and rich texture details (top three rows). Our method also supports creating high-fidelity digital avatars (the forth row) conditioned on single portrait images (visualized in dashed boxes) and high-quality 3D assets given text prompts (the fifth row).

require per-scene optimization to achieve photo-realistic quality. In contrast, our work proposes a structured representation termed GaussianCube for 3D generative tasks.

3D generation. Previous works of SDS-based optimization [44, 55, 67, 52, 12, 53, 70, 56] distill 2D diffusion priors [47] to a 3D representation with the score functions, but these works are notably time-intensive, often taking several minutes to hours. While 3D-aware GANs [8, 19, 7, 21, 42, 16, 66] facilitate view-dependent image generation from single-image collections, they struggle to capture the complexity of diverse objects with intricate geometric variations [65]. Although recent works [59, 39, 22, 57, 49, 73] have utilized diffusion models with structured proxy representations for 3D generation, the use of a shared implicit feature decoder across different assets restricts expressiveness and the computational demands of NeRF hinder efficient training. In contrast, we introduce a structured and fully explicit radiance representation for 3D generative modeling, building upon 3DGS [30]. A concurrent work of [23] includes elaborate designs to form the Gaussians into volumetric representation during fitting, yet does not thoroughly address global correspondence. In contrast, our approach only restricts the total count of Gaussians while allowing freedom in their spatial distribution during the fitting. We then organize these Gaussians into a voxel grid using Optimal Transport, which yields a spatially coherent arrangement with minimal global offset cost, effectively easing the difficulty of generative modeling.

3 Method

Following prior works, our framework comprises two primary stages as shown in Figure 2: 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 the proposed spatially structured GaussianCube

https://doi.org/10.52202/079017-3091

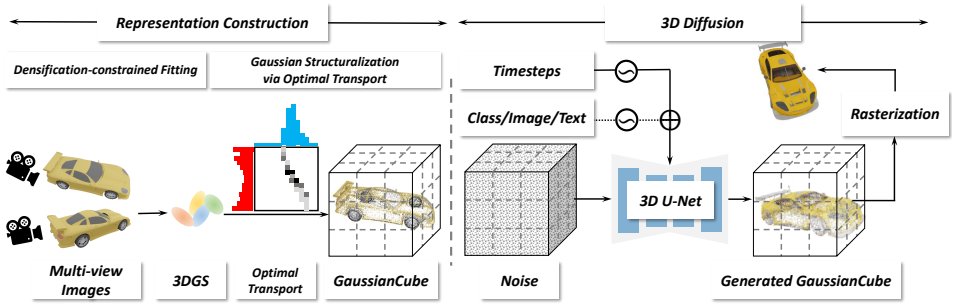


Figure 2: **Overall framework.** Our framework comprises two main stages of representation construction and 3D diffusion. In the representation construction stage, given multi-view renderings of a 3D asset, we perform *densification-constrained fitting* to obtain 3D Gaussians with constant numbers. Subsequently,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

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. We will detail our designs for each stage subsequently.

3.1 Representation Construction

We build our GaussianCube upon 3DGS, an explicit representation that offers impressive fitting quality and real-time rendering speed. However, it fails to yield Gaussians of fixed length since the adaptive density control during GS fitting can lead to a varying number of Gaussians for different objects. Furthermore, the lack of a predetermined spatial ordering for Gaussians leads to a disorganized 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 structured and explicit radiance representation.

Formally, a 3D asset is represented by a collection of 3D Gaussians as introduced in Gaussian Splatting [30]. The geometry of the i-th 3D Gaussian $\boldsymbol{g}_i$ is given by

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

where $\boldsymbol{\mu}_i \in \mathbb{R}^3$ is the center of the Gaussian and $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3\times3}$ 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 $\boldsymbol{g}_i$.

Densification-constrained fitting. Our approach begins with the aim of maintaining a constant number of Gaussians $\boldsymbol{g} \in \mathbb{R}^{N_{\text{max}}\times C}$ across different objects during the fitting. A simplistic approach might involve omitting the densification and pruning in the original GS. However, we argue that such simplifications significantly harm the fitting quality, with empirical evidence shown in Table 6. Instead, we propose to retain the pruning process while imposing a new constraint on the densification phase as shown in Figure 3 (a). The fitting process encompasses several distinct stages: 1) Densification Detection: Assuming the current iteration includes N_{c} Gaussians, we identify densification candidates by selecting those with view-space position gradient magnitudes exceeding a predefined threshold τ. We denote the number of candidates as N_d. 2) Candidate sampling: To prevent exceeding the predefined maximum of N_{max} Gaussians, we select $\min\left(N_{\text{max}} - N_{\text{c}}, N_d\right)$ Gaussians with the largest view-space positional gradients from the candidates for densification. 3) Densification: We modify the densification approach by alternating between cloning and splitting actions into separate steps. 4) Pruning Detection and Pruning: We identify and remove the Gaussians with α less than a small threshold ϵ. After completing the fitting process, we pad Gaussians with $\alpha = 0$ to reach the target count of N_{max} without affecting the rendering results. Benefiting from our proposed strategy, we

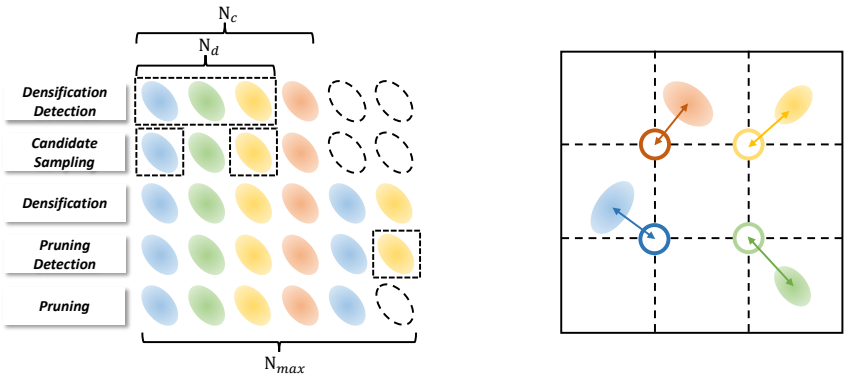


Figure 3: **Illustration of representation construction.** First, we perform densification-constrained fitting to yield a fixed number of Gaussians, as shown in (a). We then employ Optimal Transport to organize the resultant Gaussians into a voxel grid. A 2D illustration of this process is presented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

attain a high-quality representation with orders of magnitude fewer parameters compared to existing works of similar quality, which significantly reduces the modeling difficulty for the diffusion models.

Gaussian structuralization 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 while preserving their geometric relations as much as possible. While naive approaches such as nearest neighbor transport fall short in conserving these relations due to disregard for global arrangement with evidence shown in Figure 10, we formulate this as an Optimal Transport (OT) problem [58, 4] 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:

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

The solution is a bijective transport plan $\mathbf{T}^*$ that minimizes the total transport distances. We employ the Jonker-Volgenant algorithm [27] to solve the OT problem. We provide a 2D illustration in Figure 3 (b). We organize the Gaussians according to the solutions, with the j-th voxel 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 replace the original Gaussian positions with offsets of the current voxel center to reduce the solution space for diffusion models. As a result, our fitted Gaussians are systematically arranged within a voxel grid $\boldsymbol{v}$ and preserve the spatial correspondence of neighboring Gaussians, which further facilitates generative modeling.

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 $\boldsymbol{y}_t := \alpha_t \boldsymbol{y}_0 + \sigma_t \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \in \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ represents the added Gaussian noise, and α_t, σ_t constitute the noise schedule. As a result, $\boldsymbol{y}_T$ will finally reach isotropic Gaussian 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 conditional generation.

https://doi.org/10.52202/079017-3091

Table 2: Comparison with prior 3D representations of spatial structure, fitting quality, relative fitting speed (Rel. Speed) and parameter sizes on ShapeNet Car. * denotes that the implicit feature decoder is shared across different objects. All methods are evaluated at 30K iterations.

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	✓	31.78	0.0676	0.9664	0.15×	67.12
Voxels*	✓	30.25	0.0926	0.9541	0.15×	67.12
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**

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

Figure 4: Qualitative results of object fitting.

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 standard U-Net network for diffusion [41, 17] and simply replace the original 2D operators including convolution, attention, upsampling and downsampling with their 3D counterparts.

Conditioning mechanism. Our model supports a variety of condition signals to control the generation process. When performing class-conditioned diffusion modeling, we employ adaptive group normalization (AdaGN) [17] to inject the class labels into our model. For image-conditioned digital avatar creation, we leverage a pretrained vision transformer [6] to encode the conditional image into a sequence of feature tokens. We subsequently adopt cross-attention to make the model learn the correspondence between 3D activations and 2D image feature tokens following [5]. We also leverage cross-attention as our condition mechanism when creating 3D objects from text, similar to previous text-to-image diffusion models [47].

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:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{t,\boldsymbol{y}_0,\boldsymbol{\epsilon}}\left[\left\|\hat{\boldsymbol{y}}_\theta\left(\alpha_t \boldsymbol{y}_0 + \sigma_t \boldsymbol{\epsilon}, t, \boldsymbol{c}_{\text{cls}}\right) - \boldsymbol{y}_0\right\|_2^2\right], \tag{3}$$

where the condition signal $\boldsymbol{c}_{\text{cls}}$ is only needed when training conditional diffusion models. We additionally impose image-level supervision to improve the rendering quality of generated GaussianCube, which has been demonstrated to effectively enhance the perceptual details in previous works [59, 39]. Specifically, we penalize the discrepancy between the rasterized images I^t_{pred} of the model prediction at timestep t and the ground-truth images I_{gt} using:

$$\mathcal{L}_{\text{image}} = \mathbb{E}_{I^t_{\text{pred}}}\left(\sum_l \left\|\Psi^l\left(I^t_{\text{pred}}\right) - \Psi^l\left(I_{\text{gt}}\right)\right\|_2^2\right) + \mathbb{E}_{I^t_{\text{pred}}}\left(\left\|I^t_{\text{pred}} - I_{\text{gt}}\right\|_2\right), \tag{4}$$

where Ψ^l is the multi-resolution feature extracted using the pre-trained VGG [50]. Benefiting from the efficiency of both rendering speed and memory costs from GS [30], we are able to perform fast training with high-resolution renderings. Our overall training loss can be formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{simple}} + \lambda \mathcal{L}_{\text{image}}, \tag{5}$$

where λ is a balancing weight.

Table 3: Quantitative results of unconditional generation on ShapeNet Car and Chair [9] and class-conditioned generation on OmniObject3D [64].

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**

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 [64]. Our model is able to handle diverse distribution with semantically accurate results.

4 Experiments

4.1 Dataset and Metrics

To measure the expressiveness and efficiency of various 3D representations, we fit 100 objects in ShapeNet Car [9] using each representation and report the PSNR, LPIPS [75] and Structural Similarity Index Measure (SSIM) metrics when synthesizing novel views. Furthermore, we conduct experiments of single-category unconditional generation on ShapeNet [9] Car and Chair, and class-conditioned generation on real-world scanned dataset OmniObject3D [64]. We compute the FID [24] and KID [2] scores between 50K generated renderings and 50K ground-truth renderings. For image-conditioned digital avatar generation, we utilize the synthetic avatar dataset [62], which comprises highly-detailed 3D avatars created by synthetic pipeline. We assess the generation quality of 5K rendering from 500 test avatars and additionally include cosine similarity of identity embedding [15] (CSIM) to measure the ID preservation. The experiments of text-to-3D generation are performed on the large-scale challenging Objaverse dataset [14]. We numerically evaluate the text alignment quality using CLIP score [46] of 300 test prompts. All images are rendered with 512×512 resolution. For more details of data, please refer to Appendix A.1.

https://doi.org/10.52202/079017-3091

Table 4: Quantitative results of digital avatar creation conditioned on single portrait image.

Method	PSNR↑	LPIPS↓	SSIM↑	CSIM↑	FID-5K↓	KID-5K(‰)↓
Rodin w/o 2D SR	18.80	0.2842	0.7439	0.6594	32.07	24.78
Rodin	18.59	0.2821	0.7373	0.6466	20.02	9.24
Ours	**21.87**	**0.1768**	**0.7703**	**0.7821**	**8.32**	**2.67**

Reference Rodin [59] **Ours**

Figure 7: Qualitative comparison of 3D avatar creation conditioned on single frontal portraits.

4.2 Implementation Details

For GaussianCube construction, we set N_{max} to 32,768 and C to 14 across all datasets. We perform the proposed densification-constrained fitting for 30K iterations, which requires approximately 2.67 minutes on a single V100 GPU for each object. After OT-based structuralization, we obtain $32 \times 32 \times 32 \times 14$ GaussianCube for each object. The OT-based structuralization takes around 2 minutes per object on an AMD EPYC 7763v CPU. For the 3D diffusion model, we adopt the ADM U-Net network [41, 17]. 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 [41] with loss weight λ set to 10. We deploy 16 Tesla V100 GPUs for the ShapeNet Car, ShapeNet Chair, OmniObject3D, and Synthetic Avatar datasets, whereas 32 Tesla V100 GPUs are used for training on the Objaverse dataset. It takes about one week to train our model on ShapeNet Car, ShapeNet Chair, and OmniObject3D, and approximately two weeks for the Synthetic Avatar and Objaverse datasets. For more training details, please refer to Appendix A.1.

4.3 Main Results

3D fitting. We first evaluate our representation capability of object fitting against previous NeRF-based representations including Voxels [57] and Triplane [8], which are widely adopted in previous 3D generation works [8, 59, 5, 39, 57]. We set the representation size of Voxels and Triplane to $128 \times 128 \times 128 \times 32$ and $256 \times 256 \times 32$ respectively for comparable fitting quality. We also include Instant-NGP [40] and original Gaussian Splatting [30] for reference despite their unsuitability for generative modeling due to their unstructured spatial nature. As shown in Table 2, our GaussianCube outperforms all NeRF-based representations among all metrics. Figure 3 illustrates that GaussianCube can faithfully reconstruct geometry details and intricate textures. Moreover, we achieve such high-quality fitting with orders of magnitude fewer parameters than previous structured representation due to the densification-constrained fitting, showcasing our compactness. Notably, the shared implicit feature decoder in the multi-object fitting of NeRF-based methods leads to significant decreases in quality compared to single-object fitting as evidenced in Table 2. While the fully explicit nature of GS results 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 [8, 19] and Triplane diffusion models [5]. As shown in Table 3, our method surpasses all prior works in terms of both FID and KID scores and sets new records. We 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 textures.

Table 5: Quantitative results of text-to-3D creation. Inference time is measured on a single A100 GPU. While Shape-E, LGM achieve comparable CLIP scores as ours, they either utilize millions of training data or leverage 2D diffusion prior.

	DreamGaussian	VolumeDiffusion	Shap-E	LGM	Ours
CLIP Score↑	26.38	24.41	30.52	30.06	**30.56**
Inference Time (s)↓	$\sim$ 120	4.95	4.42	**1.55**	2.30

DreamGaussian [53] VolumeDiffusion [57] Shap-E [28] LGM [54] **Ours**

Figure 8: Qualitative comparison of text-to-3D generation on Objaverse [14]. Our model is able to generate high-quality samples according to the given text prompts.

Large-vocabulary class-conditioned generation. We also compare class-conditioned generation with DiffTF [5] on more diverse and challenging OmniObject3D [64] dataset. We achieve significantly better FID and KID scores than DiffTF as shown in Table 3. 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.

Image-conditioned avatar generation. For 3D avatar generation conditioned on a single reference image, we compare our method with state-of-the-art Triplane diffusion models, Rodin [47]. Our model surpasses Rodin among all evaluated metrics as shown in Table 4. Although Rodin utilizes a 2D refiner [60] to boost the visual quality of facial areas, which significantly compromises 3D consistency. Our model still outperforms it by direct real-3D generation. Results in Figure 7 demonstrate that our model faithfully preserves the identity, expression and accessories of the references with rich details, while Rodin struggles to provide satisfactory results even using 2D refinement.

Text-to-3D generation. We compare text-to-3D generation with prior arts including diffusion models [28, 57], optimization-based method [53] and feed-forward method [54]. Our model achieves competitive text-3D alignment results as shown in Table 5. The visual comparison in Figure 8 shows that our model is able to create high-quality samples aligning with text prompts in just 2.3 seconds. DreamGaussian tends to create over-saturated results and suffers from Janus problem. VolumeDiffusion produces unsatisfactory textures with poor text alignment. Shap-E can produce semantically accurate results but struggles to generate complex geometry. LGM reconstructs 3D Gaussians from multi-view images generated by text-conditioned multi-view diffusion pipeline [48], whereas the inconsistency [54] of the generated multi-views often results in inaccurate geometric reconstruction.

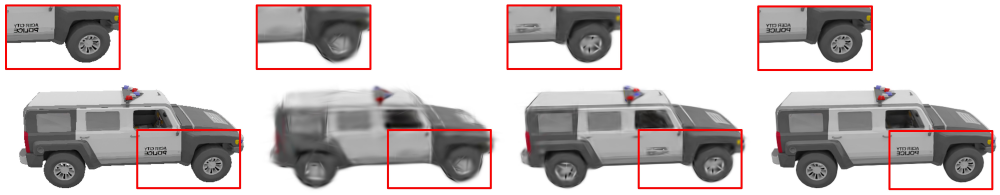
4.4 Ablation Study

We first examine the key factors in representation construction on ShapeNet Car. To spatially structure the Gaussians, a simplistic 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 9. Even by introducing learnable offsets to the voxel grid, the results still lack details. We observe the offsets are typically too small to effectively lead the Gaussians close to the object surfaces, which indicates the importance of densification in the fitting process. Instead, GaussianCube can capture both complex geometry and intricate details as shown in Figure 9. The numerical comparison in Table 6 also demonstrates the superior fitting quality of GaussianCube.

https://doi.org/10.52202/079017-3091

Table 6: Quantitative ablation of both representation fitting and generation quality on ShapeNet Car.

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**



Ground-truth Table 6 A. Table 6 B. **Table 6 D. (Ours)**

Figure 9: Qualitative ablation of representation fitting.

OT (Ours) Nearest Neighbor Table 6 B. Table 6 C. **Table 6 D. (Ours)**

(a) (b)

Figure 10: Visual ablation of the Gaussian organization methods and 3D generation. For visualization of Gaussian structuralization in (a), we map the coordinates of the corresponding voxel of each Gaussians to RGB values to visualize the organization. Our OT-based solution also results in the best generation quality shown in (b).

We also evaluate how the representation affects 3D generative modeling on ShapeNet Car as shown in Table 6 and Figure 10. Limited by the poor fitting quality, performing diffusion modeling on voxel grid with learnable offsets leads to blurry generation results as shown in Figure 10. To validate the importance of organizing Gaussians via Optimal Transport (OT), we compare with the organization based on nearest neighbor transport. We linearly map each Gaussian's corresponding coordinates of voxel to RGB color to visualize different organizations. As shown in Figure 10 (a), our proposed OT approach yields smooth color transitions, indicating that our method successfully preserves the spatial correspondence. However, nearest neighbor results in abrupt color transitions due to their disregard for global structure. Both the quantitative results in Table 6 and visual comparisons Figure 10 indicate that our globally structured arrangement facilitates generative modeling by alleviating its complexity, successfully leading to superior generation quality.

5 Conclusion

We have presented GaussianCube, a structured and explicit radiance representation crafted for 3D generative models. We begin by fitting each 3D object with a constant number of Gaussians using our proposed densification-constrained fitting algorithm. We further organize the obtained Gaussians into a spatially structured representation by solving the Optimal Transport between the positions of Gaussians and the predefined voxel grid. The proposed GaussianCube is spatially structured, allowing to use standard 3D U-Net for diffusion modeling without elaborate designs. Moreover, GaussianCube can achieve high-quality fitting using much fewer parameters compared to prior works of similar quality, which further eases the difficulty of generative modeling. Our 3D diffusion models equipped with GaussianCube achieve state-of-the-art generation quality on the evaluated datasets, underscoring its potential of GaussianCube as a versatile and powerful radiance representation for 3D generation.

Acknowledgments: This work was supported in part by the Anhui Provincial Natural Science Foundation under Grant 2108085UD12. We acknowledge the support of GPU cluster built by MCC Lab of Information Science and Technology Institution, USTC. We also thank anonymous reviewers for their valuable comments.

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] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023.

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

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

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021.

[7] 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.

[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, 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.

[9] 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.

[10] Guikun Chen and Wenguan Wang. A survey on 3d gaussian splatting. *arXiv preprint arXiv:2401.03890*, 2024.

[11] Hansheng Chen, Jiatao Gu, Anpei Chen, Wei Tian, Zhuowen Tu, Lingjie Liu, and Hao Su. Single-stage diffusion nerf: A unified approach to 3d generation and reconstruction. *arXiv preprint arXiv:2304.06714*, 2023.

[12] 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.

[13] 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.

[14] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023.

[15] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[16] 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.

 https://doi.org/10.52202/079017-3091

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

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

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

[26] 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.

[27] 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.

[28] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

[29] 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.

[30] 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.

[31] 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.

[32] Zhengqi Li, Richard Tucker, Noah Snavely, and Aleksander Holynski. Generative image dynamics. *arXiv preprint arXiv:2309.07906*, 2023.

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations, ICLR*, 2019.

[34] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

[35] Guanxing Lu, Shiyi Zhang, Ziwei Wang, Changliu Liu, Jiwen Lu, and Yansong Tang. Manigaussian: Dynamic gaussian splatting for multi-task robotic manipulation. *arXiv preprint arXiv:2403.08321*, 2024.

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

[37] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

[38] 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.

[39] 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.

[40] 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.

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

[42] 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.

[43] 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.

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

[45] 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.

[46] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[47] 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.

[48] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. *arXiv preprint arXiv:2308.16512*, 2023.

[49] 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.

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

[51] 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.

 https://doi.org/10.52202/079017-3091

[52] 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.

[53] 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.

[54] 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.

[55] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22819–22829, 2023.

[56] Junshu Tang, Yanhong Zeng, Ke Fan, Xuheng Wang, Bo Dai, Kai Chen, and Lizhuang Ma. Make-it-vivid: Dressing your animatable biped cartoon characters from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6243–6253, 2024.

[57] 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.

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

[59] 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.

[60] Xintao Wang, Yu Li, Honglun Zhang, and Ying Shan. Towards real-world blind face restoration with generative facial prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9168–9178, 2021.

[61] Zhenwei Wang, Tengfei Wang, Zexin He, Gerhard Hancke, Ziwei Liu, and Rynson WH Lau. Phidias: A generative model for creating 3d content from text, image, and 3d conditions with reference-augmented diffusion. *arXiv preprint arXiv:2409.11406*, 2024.

[62] Erroll Wood, Tadas Baltrušaitis, Charlie Hewitt, Sebastian Dziadzio, Thomas J Cashman, and Jamie Shotton. Fake it till you make it: face analysis in the wild using synthetic data alone. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3681–3691, 2021.

[63] 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.

[64] 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.

[65] Weihao Xia and Jing-Hao Xue. A survey on deep generative 3d-aware image synthesis. *ACM Computing Surveys*, 56(4):1–34, 2023.

[66] 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.

[67] 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.

[68] 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.

[69] 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.

[70] 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.

[71] Youyi Zhan, Tianjia Shao, He Wang, Yin Yang, and Kun Zhou. Interactive rendering of relightable and animatable gaussian avatars. *arXiv preprint arXiv:2407.10707*, 2024.

[72] 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.

[73] Bowen Zhang, Yiji Cheng, Chunyu Wang, Ting Zhang, Jiaolong Yang, Yansong Tang, Feng Zhao, Dong Chen, and Baining Guo. Rodinhd: High-fidelity 3d avatar generation with diffusion models. *arXiv preprint arXiv:2407.06938*, 2024.

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

[75] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[76] Junsheng Zhou, Weiqi Zhang, and Yu-Shen Liu. Diffgs: Functional gaussian splatting diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[77] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5826–5835, 2021.

 https://doi.org/10.52202/079017-3091

A Appendix

A.1 Additional Implementation Details

Dataset preparation. We conduct experiments on ShapeNet Car [9], ShapeNet Chair [9], OmniObject3D [64], Synthetic Avatar [62] and Objaverse [14] datasets. For each dataset, we report the total number of objects used for training, the number of views rendered per object for GaussianCube fitting and the distribution of camera poses used for rendering in Table 7. For the Objaverse dataset, we excluded low-quality objects, such as those without textures or with defective reconstructions following [57]. We also report the object bounding box $\boldsymbol{b}$ in the world coordinate system of each dataset in Table 7, which is used to construct the predefined voxel grid within $[-\boldsymbol{b}, \boldsymbol{b}]^3$ during OT-based Gaussian structuralization.

Representation construction. We set N_{max} to 32768 and C to 14 omitting the view-dependent spherical harmonics. This simplification appears to have a negligible impact on object fitting while concurrently reducing the data dimension, thereby alleviating the difficulty of diffusion modeling. During our densification-constrained fitting procedure, we primarily follow the hyper-parameters in original Gaussian Splatting [30]. For OT-based Gaussian structuralization, we adopt an approximate solution for the OT problem due to the $O\left(N_{\text{max}}^3\right)$ time complexity of Jonker-Volgenant algorithm [27]. 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 find this approximation successfully strikes a balance between computational efficiency and spatial structure preservation. The proposed densification-constrained fitting takes around 2.67 minutes for each object of 30K iterations and the OT-based voxelization takes around 2 minutes which can be run on CPU in parallel.

3D Diffusion. To train the 3D diffusion model, we initially compute the instance-wise statistics of mean $\bar{\boldsymbol{\mu}} \in \mathbb{R}^{N_v \times N_v \times N_v \times C}$ and standard deviation $\bar{\boldsymbol{\sigma}} \in \mathbb{R}^{N_v \times N_v \times N_v \times C}$, from the GaussianCubes of each training dataset respectively. These statistical measures are then utilized to normalize the training data. For our 3D diffusion model architecture, we adopt the ADM-UNet from [17] and replace the convolution, upsampling, downsampling and attention operations with 3D implementations. We train our model using AdamW optimizer [33], and apply exponential moving average (EMA) with a rate of 0.9999 during training. We clamp the prediction of opacity α to $[0, 1)$ and clamp the minimum value of predicted scaling $\boldsymbol{s}$ to 0 to ensure validity. For unconditional generation on ShapeNet, we train the model with a base learning rate $5e-5$ for 850K iterations and then decay the learning rate to $5e-6$ for another 150K iterations. For 3D digital avatar creation from a single portrait image, we adopt the pretrained DINO ViT-B/16 [6] to encode the 512×512 conditional images into 1025×768 conditional feature tokens. For text-to-3D creation, we take CLIP-L/14 [46] to encode the text prompts into 77×768 conditional feature tokens. We provide more detailed configurations of the model architectures, diffusion training and inference for each dataset in Table 8.

Implementation of Gaussian organization visualization in Figure 10 (a). For the i-th Gaussian, we obtain its corresponding voxel grid centers $\boldsymbol{x}_k \in \mathbb{R}^3$ according to Optimal Transport plan $\mathbf{T}^*$ (*i.e.*, $\mathbf{T}^*_{ik} = 1$) as illustrated in Section 3.1. To visualize the coordinates of $\boldsymbol{x}_k$, we map them to RGB color $\boldsymbol{C}_k \in \mathbb{R}^3$ using:

$$\boldsymbol{C}_k = \frac{(\boldsymbol{x}_k + \boldsymbol{b})}{2\boldsymbol{b}} \times \mathbf{255}, \tag{6}$$

where $\boldsymbol{b}$ is the bounding box in the world coordinate system. The resultant point cloud like visualizations are shown in Figure 10 (a), where smooth color transitions indicate coherent spatial correspondence preservation.

A.2 Additional Ablation Study and Analysis

Ablation of N_{max} in densification-constrained fitting. We conduct experiments to evaluate how N_{max} affects fitting on ShapeNet Car. The results in Table 9 indicate that there is a clear trend where increasing N_{max} leads to improved fitting accuracy. However, a larger N_{max} also incurs higher computational costs during diffusion training. Therefore, we set N_{max} to 32,768 to strike a balance between high-quality fitting and computational efficiency.

Table 7: Details of each dataset.

Dataset	# Objects	# Views per object	Rotation Angle	Elevation Angle	Bounding Box
ShapeNet Car	7,462	150	$[0, 2\pi]$	$[\frac{1}{6}\pi, \frac{1}{2}\pi]$	0.45
ShapeNet Chair	6,775	150	$[0, 2\pi]$	$[\frac{1}{6}\pi, \frac{1}{2}\pi]$	0.35
OmniObject3D	5,795	100	$[0, 2\pi]$	$[0, \frac{1}{2}\pi]$	1.0
Synthetic Avatar	98,000	300	$[0, 2\pi]$	$[\frac{1}{6}\pi, \frac{2}{3}\pi]$	40.0
Objaverse	125,653	150	$[0, 2\pi]$	$[0, \frac{2}{3}\pi]$	0.5

Table 8: Detailed configuration of model architecture, diffusion training and inference on each dataset.

	ShapeNet Car	ShapeNet Car	OmniObject3D	Synthetic Avatar	Objaverse
Diffusion Steps	1,000	1,000	1,000	1,000	1,000
Noise Schedule	Cosine	Cosine	Cosine	Cosine	Cosine
NFEs	300	300	300	250	44
Inference Time (s)	10.06	10.06	10.06	13.80	2.30
Inference Sampler	DPM-solver [34]	DPM-solver [34]	DPM-solver [34]	DPM-solver [34]	DPM-solver [34]
DPM-solver Order	3	3	3	2	2
DPM-solver Mode	Multi-step	Multi-step	Multi-step	Multi-step	Adaptive
CFG Scale	-	-	2.0	1.3	3.5
Model Size	82M	82M	82M	339M	339M
Channels	64	64	64	128	128
Channel Mult.	(1,2,3,4)	(1,2,3,4)	(1,2,3,4)	(1,2,3,4)	(1,2,3,4)
Num. Res. Blocks	3	3	3	3	3
Attn Resolutions	(8, 4)	(8, 4)	(8, 4)	(8, 4)	(8, 4)
Num. Head Channels	64	64	64	64	64
Dropout	0	0	0	0	0
Scale Shift Norm	True	True	True	True	True
Training Steps	1,000K	1,000K	700K	1,200K	1,800K
Training GPUs	16	16	16	16	32
Batch Size	128	128	128	128	256
Base Lr	$5e-5$	$5e-5$	$5e-5$	$5e-5$	$5e-5$
Lr Decay Steps	850K	850K	-	-	-

Ablation of classifier-free guidance in class-conditioned generation. We study how classifier-free guidance (CFG) impacts our generation quality when inference class-conditioned diffusion models. We report the FID and KID metrics in Table 10 under different CFG scales.

Visualization of intermediate results in the denoising process. During inference, our model starts from Gaussian noise and progressively denoises to yield the high-quality GaussianCube. We present visualizations of the intermediate renderings $\boldsymbol{y}_t$ at various timesteps $t \in [0, T]$ throughout the denoising process, offering a detailed insight into the GaussianCube diffusion procedure. As illustrated in Figure 11, our model first establishes the global structure and then incrementally enhances the details, which is similar to previous 3D diffusion models [59, 49].

Nearest neighbors analysis. We perform nearest neighbor search of some unconditionally generated samples in the paper according to the similarity of pretrained CLIP [46] features. The results in Figure 12 demonstrate that our model is capable of generating novel geometry and textures rather than simply memorizing the training data.

Distribution visualization of offset from voxel grids of fitted GaussianCubes. We visualize the offset distribution of 1K randomly selected GaussianCubes from each experimental dataset in Figure 14. We observe that most distributions exhibit a bell curve, similar to a normal distribution. However, the Digital Synthetic Avatar dataset presents a more uniform distribution with multiple peaks. We believe these distributions offer valuable insights into how well the fitted 3D Gaussians align with voxel grid centers. Bell-shaped distributions akin to a normal distribution, such as in the ShapeNet Car and Chair datasets, suggest a strong initial alignment and lower complexity. On the other hand, broader distributions (e.g., the Digital Synthetic Avatar dataset) indicate a higher level of detail (for instance, hair) and a greater need for adjustments during organization.

A.3 Additional Visual Results

For 3D avatar generation, while trained on synthetic dataset, our model is capable of generalizing to in-the-wild portrait input. We provide more visual comparison of 3D avatar creation conditioned on in-

https://doi.org/10.52202/079017-3091

Table 9: Quantitative ablation of N_{max} in densification-constrained fitting. We set N_{max} to 32,768 in this paper.

N_{max}	N_v	PSNR↑	LPIPS↓	SSIM↑
4096	16	32.56	0.0547	0.9765
13824	24	34.32	0.0396	0.9842
32768	32	34.94	0.0347	0.9863
110592	48	35.29	0.0307	0.9874
262144	64	35.34	0.0301	0.9875

Table 10: Quantitative ablation of CFG scale in the class-conditioned generation of OmniObject3D [64].

Scale	w/o CFG	1.3	1.5	2.0	3.0	6.0
FID-50K↓	13.39	12.07	11.72	**11.62**	12.99	32.80
KID-50K(‰)↓	4.01	3.12	3.00	**2.78**	3.17	14.36

the-wild portraits with Rodin [59] in Figure 15. We also include additional comparison conditioned on synthetic input from our test in Figure 16. Our model can faithfully retain the identity of the reference portrait and is able to provide high-fidelity results with rich details, *e.g.* hair, glasses and clothing. Although utilizing a pretrained 2D super-resolution module which significantly compromises 3D consistency, Rodin struggles to follow the conditional images and fails to produce detailed textures in non-facial areas *e.g.* clothing and hair.

We include additional qualitative comparison and generated samples of text-to-3D generation in Figure 17 and Figure 18 respectively. Our model yields samples with better visual quality, and is capable of handling challenging prompts. The results in Figure 19 show the generation diversity of our results given the same text prompt. Our model is also capable of performing text-guided editing of generated objects by leveraging SDEdit [37] as depicted in Figure 20, demonstrating the promise of achieving controllable 3D generation.

We provide more generated samples of unconditional and class-conditioned generation in Figure 21, Figure 22 and Figure 23. The additional results demonstrate the strong capability of our model to create high-quality 3D assets with complex geometry and intricate textures.

Furthermore, we also provide an additional video in supplementary material, which intuitively illustrates our approach and visualizes the generated results.

A.4 Limitations

While GaussianCube represents a substantial step forward in developing an ideal representation for 3D content generation, it still has some limitations. Specifically, although the GaussianCube construction procedure is considerably more rapid than that of NeRF-based methods and can be executed in parallel, it still requires approximately 5 minutes to construct each object. This presents a challenge for scaling up training on extensive 3D datasets. In future work, we plan to investigate more time-efficient methods for GaussianCube construction. Additionally, akin to prior 2D diffusion models, our text-to-3D diffusion model encounters difficulties in presenting the specified number of objects within prompts as shown in Figure 13. To address this, we will look into enhancing the precision and controllability of 3D generation in the future.

A.5 Broader Impacts

The proposed GaussianCube enables high-quality 3D asset fitting with few parameters, which significantly simplifies the challenges of 3D generative modeling. Our diffusion model is capable of generating high-quality 3D assets of complex geometry and intricate textures while also accommodating a variety of conditional signals to steer the creating procedure. The strong capability of GaussianCube suggests its potential to serve as a versatile 3D representation for a variety of applications in future 3D research endeavors.

Like all generative models, particular caution is required when dealing with sensitive tasks involving human representations. Our avatar creation model is trained exclusively on a synthetic dataset [62] composed of large-scale 3D digital avatars which are generated through a graphics pipeline. We conceptualize digital avatars as analogous to those created by specialized 3D artists, rather than

photorealistic human images. This strategy in selecting training data mitigates privacy and copyright issues that might arise from utilizing real human photo collections. Nevertheless, it is crucial to acknowledge that avatars generated by our model from real-world imagery could still be misused for spreading disinformation. As such, we advocate implementing rigorous safeguards and promoting responsible use of our technology other related ones to mitigate such risks.

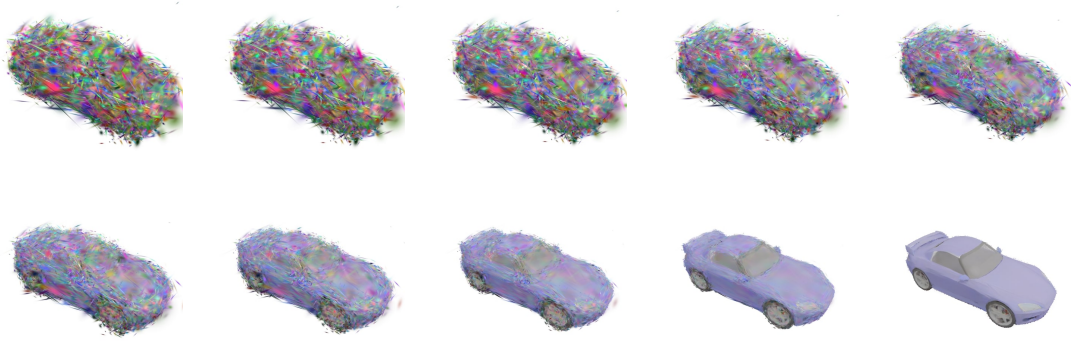


Figure 11: Visualization of generation results in intermediate diffusion timesteps.

Figure 12: Visualization of nearest neighbor search on ShapeNet Car and Chair.

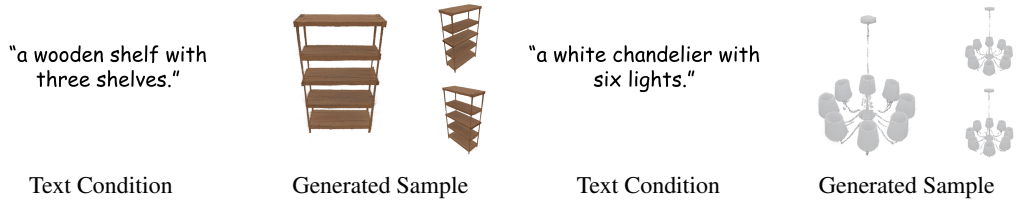


Figure 13: Failure cases.

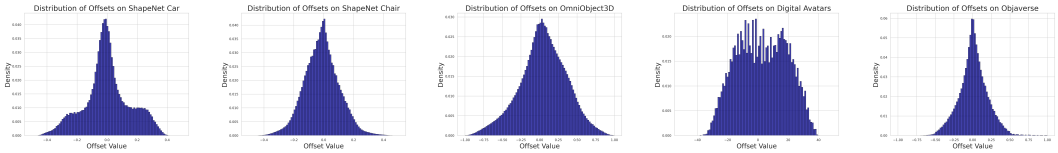


Figure 14: Distribution of offsets from voxel centers in a random selection of 1K GaussianCubes on each experimental dataset.

https://doi.org/10.52202/079017-3091

Reference Rodin [59] **Ours**

Figure 15: Additional qualitative comparison of 3D avatars creation conditioned on single in-the-wild portraits.

Reference Rodin [59] **Ours**

Figure 16: Qualitative comparison generated digital avatars conditioned on synthetic portraits.

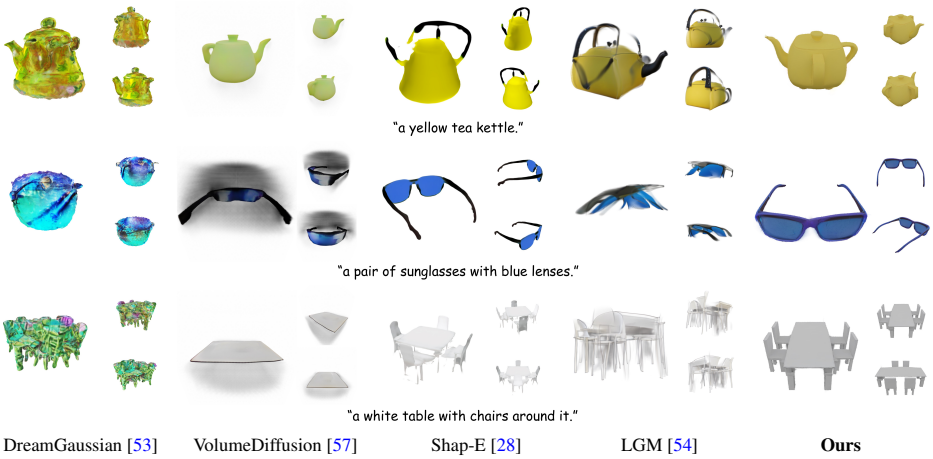


Figure 17: Additional qualitative comparison of text-to-3D generation on Objaverse [14]. Our model is capable of creating high-quality samples following input text prompts.

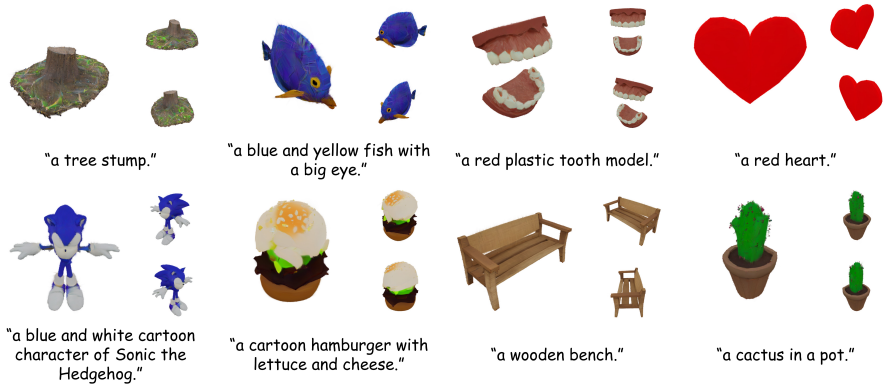


Figure 18: Additional results of text-to-3D generation.

Figure 19: Variation of text-to-3D generation. Our model is able to generate diverse results conditioned on the same text prompt.

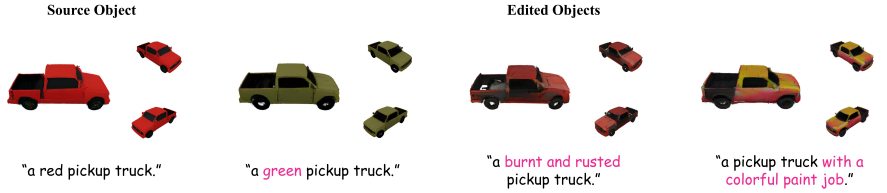


Figure 20: Example of text-guided 3D editing.

https://doi.org/10.52202/079017-3091

Figure 21: Additional generated samples on ShapeNet Car.

Figure 22: Additional generated samples on ShapeNet Chair.

https://doi.org/10.52202/079017-3091

Figure 23: Additional generated samples on OmniObject3D.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We include the discussion of limitations in Appendix A.4.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

https://doi.org/10.52202/079017-3091

Justification: this paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have provided comprehensive implementation details in the main paper and supplementary materials to ensure that our method can be faithfully reproduced. We also release our code and model checkpoints at https://github.com/GaussianCube/GaussianCube.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We have released the source code and model checkpoints at https://github.com/GaussianCube/GaussianCube.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We include comprehensive training and test details in Appendix A.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The results are accompanied by statistical significance tests.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

https://doi.org/10.52202/079017-3091

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We include sufficient information on the computer resources needed to reproduce the experiments in Appendix A.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader impacts in Appendix A.5.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We also include the safeguards in Appendix A.5.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The creators or original owners of assets are properly credited in this paper. The license and terms of use are explicitly mentioned and properly respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

https://doi.org/10.52202/079017-3091

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: This paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-3091