GAP: Gaussianize Any Point Clouds with Text Guidance

Anonymous ICCV submission

Paper ID 2942

Figure 1. GAP gaussianizes point clouds into high-fidelity 3D Gaussians with diverse appearances. Left: Examples of text-guided Gaussian generation from object-level point cloud. Bottom-right: Scene-level results with prompts 'A modern lounge' and 'A rainbow bedroom'.

Abstract

3D Gaussian Splatting (3DGS) has demonstrated its advantages in achieving fast and high-quality rendering. As point clouds serve as a widely-used and easily accessible form of 3D representation, bridging the gap between point clouds and Gaussians becomes increasingly important. Recent studies have explored how to convert the colored points into Gaussians, but directly generating Gaussians from colorless 3D point clouds remains an unsolved challenge. In this paper, we propose GAP, a novel approach that gaussianizes raw point clouds into high-fidelity 3D Gaussians with text guidance. Our key idea is to design a multi-view optimization framework that leverages a depth-aware image diffusion model to synthesize consistent appearances across different viewpoints. To ensure geometric accuracy, we introduce a surface-anchoring mechanism that effectively constrains Gaussians to lie on the surfaces of 3D shapes during optimization. Furthermore, GAP incorporates a diffuse-based inpainting strategy that specifically targets at completing hard-to-observe regions. We evaluate GAP on the Point-to-Gaussian generation task across varying complexity levels, from synthetic point clouds to challenging real-world scans, and even large-scale scenes.

1. Introduction

Point clouds serve as a fundamental representation in 3D computer vision, playing a crucial role across various domains, e.g., autonomous driving, augmented/virtual reality and robotics. With recent advances in 3D scanning devices, such as LiDAR sensors and depth cameras, point clouds have bridged the gap between the physical and digital worlds. However, it still remains a research challenge to effectively transform the geometries of raw point clouds into high-quality 3D appearances that maintain structural fidelity while providing visual-appealing renderings.

For high-quality 3D visualization, mesh-based representation has long been the standard approach. However, such a representation faces two major limitations: (1) for meshes with dense faces, the constrained texture resolution limits the final rendering quality, and (2) the heavy reliance on UV unwrapping [58] introduces additional complications such as texture overlapping, fragmentation, and distortion issues. While these limitations can be addressed with careful manual intervention, they present significant obstacles in fully automated pipelines. Recent advances in 3D Gaussian Splatting (3DGS) [21] have revolutionized neural rendering by offering an efficient and high-quality alternative to NeRF-based [27] or mesh-based representations. Moreover, 3DGS eliminates the need for explicit UV parameterization, which makes it particularly attractive for real-world applications.

Althought several attempts have been made to bridge point clouds and 3DGS, existing approaches still face several significant limitations. For example, Large Point-to-Gaussian [25] model trains a feedforward network for Gaussian primitive generation, but it requires point cloud inputs with color attributes. DiffGS [57] approaches this challenge by learning a reconstruction scheme from points to Gaussians, yet struggles in generalizing to generate diverse and high-quality 3D appearances.

To address these challenges, we propose GAP, a novel approach that generates high-quality Gaussian primitives by Gaussianizing 3D raw point clouds. GAP leverages both geometric information from input point clouds and appearance guidance from pretrained text-to-image diffusion models. Specifically, we first introduce a progressive generation scheme that optimizes Gaussian primitives across multiple viewpoints by leveraging a depth-aware text-to-image diffusion model. To ensure geometric accuracy, we design a surface-anchoring mechanism that effectively constrains Gaussians to lie on object surfaces during optimization, leading to Gaussian generations consistent to the geometry. After optimization, the generated high-quality Gaussians can cover most of the surface, however, there are still some unseen areas that require further processing. To address this, we propose a diffuse-based Gaussian inpainting strategy that gaussianizes the unseen points by leveraging the spatial relationships and geometric consistency of the visible Gaussians. To this end, GAP generates high-fidelity 3D Gaussians that maintain both geometric accuracy and visual quality.

We evaluate GAP extensively across diverse datasets, including both synthetic and real-world scanned point clouds of objects and scenes. Comprehensive experiments demonstrate that our method consistently outperforms state-of-the-art alternatives in terms of visual quality. We believe GAP opens new possibilities for Point-to-Gaussian generation, bridging the gap between widely-used, easily accessible point cloud data and high-quality 3D Gaussian representations. Our contributions can be summarized as follows:

- We proposed GAP, a novel framework that gaussianizes raw point clouds into high-quality Gaussian primitives. GAP introduces both geometric priors and text guidance with large text-to-image diffusion models to generate diverse and visual-appealing appearances from point clouds.
- We design a Gaussian optimization framework that progressively optimizes Gaussian attributes across multiple viewpoints, with a surface anchoring constraint to ensure geometric accuracy. A diffuse-based Gaussian inpainting strategy is further introduced to handle occluded regions.
- Comprehensive evaluations under synthetic and real-scanned point cloud datasets of objects and scenes demonstrate that GAP significantly outperforms the state-of-the-art methods.

2. Formatting your paper

2.1. Texture Generation

The advent of deep learning has revolutionized texture generation for 3D models. Early learning-based approaches primarily utilized GANs [15, 28, 31], while recent methods [5, 20, 24, 43, 48] leverage large-scale text-to-image diffusion models [16, 35] as powerful priors for high-fidelity texture synthesis. A series of works [9, 26, 45] adopts Score Distillation Sampling [30] as their optimization strategy for texture generation, iteratively refining textures through optimizing rendered images with respect to text prompts. Another stream of research [7, 33, 39] proposes efficient texture synthesis through depth-guided inpainting, where textures are progressively generated along specified viewpoints. Additionally, some approaches [1, 10, 51] focus on multi-view generation with geometric guidance, followed by UV-space refinement. However, maintaining texture continuity across UV seams remains challenging due to the discontinuous nature of UV mapping. Despite these advances, UV distortion and cross-view consistency remain challenging, particularly for complex objects.

2.2. Rendering-Driven 3D Representation

While mesh-based representations [4, 37] remain the standard for 3D visualization, they face limitations in texture resolution and UV parameterization [22, 36]. Remarkable progress has been achieved in the field of novel view synthesis with the proposal of Neural Radiance Fields (NeRF) [27]. Through volume rendering[13] optimization, NeRF achieves outstanding view synthesis quality, though its computational overhead during rendering is considerable. 3D Gaussian Splatting (3DGS) has emerged as an advanced 3D representation which shows convincing performance in real-time rendering. [21, 23, 38, 42, 46, 50] By representing scenes with a set of 3D Gaussian primitives, 3DGS achieves both high-quality rendering and efficient real-time performance.

2.3. 3DGS Generation Methods

With the advancement of 3D Gaussian Splatting, developing effective generative models for 3DGS has emerged as a popular research topic. A series of studies [17, 44, 52, 61] have explored image-based reconstruction without generative modeling, which fundamentally limits their ability to generate diverse shapes. These methods also lack point-conditioned generation capabilities. Recent point cloud-to-Gaussian conversion approaches [25] rely heavily on the availability of RGB point clouds as input. While Gaussian painter [56] uses reference images for stylization, it lacks precise control over the final appearance. This highlights the need for a framework generating high-quality Gaussians from point clouds with flexible appearance control.

3. Method

We introduced GAP, a novel method that establishes a bridge between raw point clouds and 3D Gaussians by neural gaussianizing. Given an input point cloud $P = \{p_i\}_{i=1}^{N}$, our goal is to generate Gaussians $G = \{g_i\}_{i=1}^{M}$ from P, conditioned on the text prompt c. The overview of GAP is shown in Fig. 2. We begin by previewing Gaussian Splatting, along with the initialization strategy in Sec. 3.1. In Sec. 3.2, we present a progressive Gaussian generation scheme that utilizes a powerful text-to-image diffusion model to generate or inpaint images from a given viewpoint. We further introduce a Gaussian optimization strategy which learns Gaussian attributes from the generated images representing high-fidelity appearance, in Sec. 3.3. While the object is largely observable from various viewpoints, certain regions remain difficult to capture. To address this, we introduce a diffuse-based Gaussian inpainting method in Sec. 3.4.

3.1. Gaussian Initialization

Preview 3D Gaussian Splatting. 3D Gaussian Splatting (3DGS) [21] is a modern representation technique that models 3D shapes or scenes through a collection of Gaussian primitives. Each Gaussian g_i is defined by a set of parameters that characterize its geometry and appearance properties. The geometry of g_i is mathematically defined by its center position $\sigma_i \in \mathbb{R}^3$ and a covariance matrix Σ_i, formulated as:

$$g_i(x) = \exp\left(-\frac{1}{2}(x-\sigma_i)^T\Sigma^{-1}(x-\sigma_i)\right). \qquad (1)$$

The covariance matrix Σ_i is constructed from a rotation matrix $r_i \in \mathbb{R}^4$ and a scale matrix $s_i \in \mathbb{R}^3$ ($\Sigma_i = r_i s_i s_i^T r_i^T$). Σ_i determines the Gaussian's shape, orientation, and range in space. Beyond geometry, each Gaussian encompasses visual attributes including an opacity term o_i and view-dependent color properties c_i, implemented as spherical harmonics.

Initialization Scheme. When generating Gaussians from an input point cloud $P = \{p_i\}_{i=1}^{N}$, we initialize the center positions σ_i of Gaussian primitives as the spatial coordinates of the points. This direct spatial mapping provides fine initial geometries for Gaussians, which roughly represent the underlying 3D surfaces. To better exploit the inherent geometric information embedded in the point cloud, we employ CAP-UDF [55] to learn a neural Unsigned Distance Field (UDF) [11] f_u from the point cloud and derive point normals $N = \{n_i\}_{i=1}^{N}$ through gradient inference:

$$n_i = \frac{\nabla f_u(p_i)}{\|\nabla f_u(p_i)\|}. \qquad (2)$$

Instead of vanilla 3DGS, we adopt 2D Gaussian Splatting (2DSG) [19] as our representation. The key idea of 2DGS is to replace 3D Gaussian ellipsoids with 2D-oriented Gaussian disks for scene representation, demonstrating better performances in representing detailed local geometries. 2DGS inherently encodes the normal as the disk orientation. We initialize the rotation matrix r_i of each Gaussian using its normal n_i from the field f_u, ensuring that each 2D Gaussian disk is accurately aligned to the correct orientation, providing a good initialization for subsequent optimization.

3.2. Multi-View Inpainting and Updating

For a sequence of specified viewpoints $\{v_j\}_{j=1}^{K}$, we progressively generate the visual appearance at each perspective to optimize the associated Gaussians. Using the learned UDF field, we employ ray marching techniques to compute the depth value for each pixel on the depth map D_j. As shown in Fig. 2(a), we render an image I_j from a specific viewpoint v_j. The rendered image I_j, along with its corresponding depth map D_j, mask M_j and text prompt c, are fed into the depth-aware inpainting model.

Depth-aware Inpainting Model. We leverage a depth-aware inpainting diffusion model [34, 53] as the appearance generation model. By integrating depth information into the diffusion-based inpainting process, the model enables more geometrically consistent image generation. Its encoder $\mathbb{E}$ operates by first encoding the masked image I concatenated with the depth map D into a latent code z_0. The initial encoding is:

$$z_0 = \mathbb{E}\left[I \parallel D\right]. \qquad (3)$$

The process gradually degrades the initial latent code through a series of noise-adding operations. At each

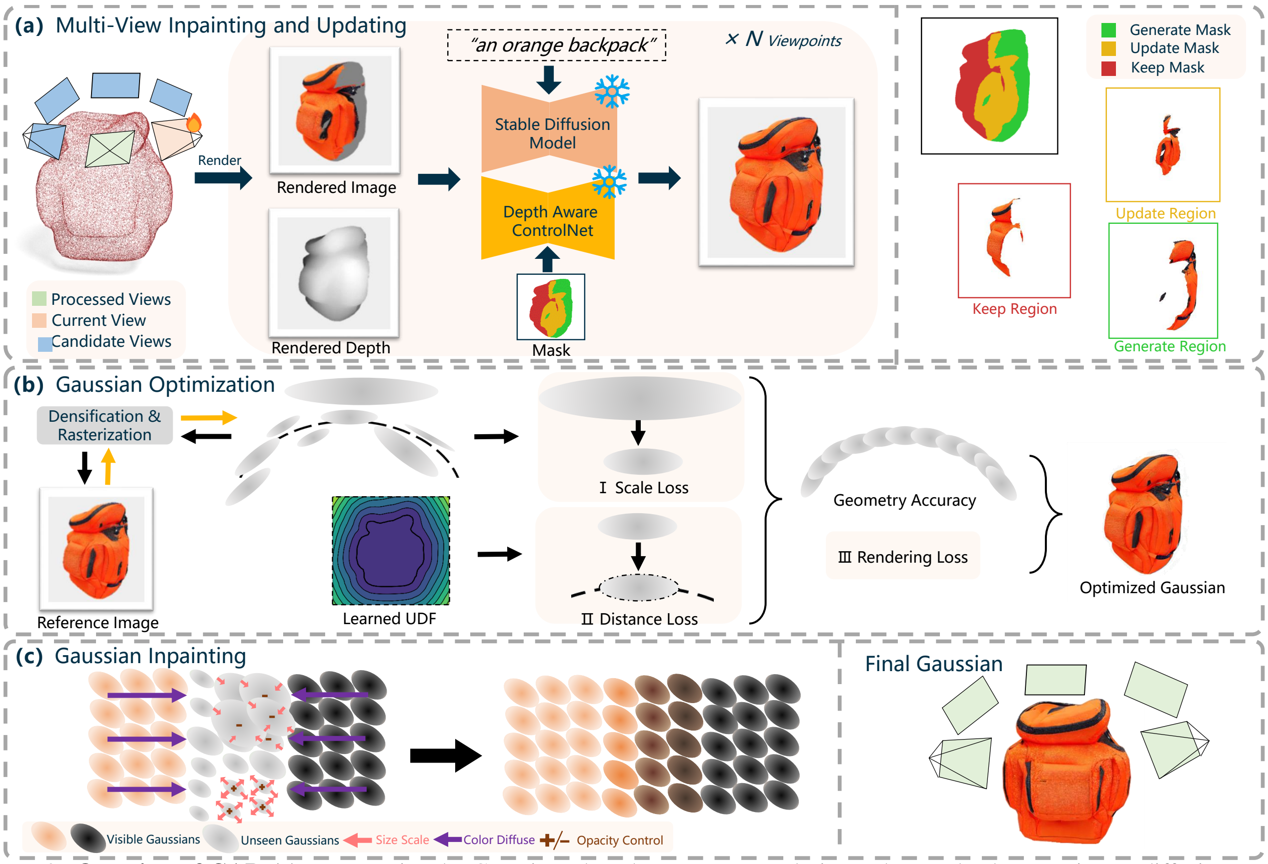

Figure 2. **Overview of GAP. (a)** We rasterize the Gaussians though an unprocessed view, where a depth-aware image diffusion model is used to generate consistent appearances using the rendered depth and mask with text guidance. The mask is dynamically classified as generate, keep, or update based on viewing conditions. **(b)** The Gaussian optimization includes three constraints: the Distance Loss and Scale Loss introduced to ensure geometric accuracy, and the Rendering Loss that ensures high-quality appearance. **(c)** The Gaussian inpainting strategy which diffuses the geometric and appearance information from visible regions to hard-to-observe areas, considering local density, spatial proximity and normal consistency.

timestep t, the model add Gaussian noise according to a variance schedule defined by β_t. The transformation follows a probabilistic distribution:

$$z_t \mid y,\, g_\phi\left(y, t, I \parallel D\right) \sim \mathcal{N}\left(\sqrt{1-\beta_t}\, z_{t-1},\ \beta_t \mathbf{I}\right), \quad (4)$$

where y is text embeddings, and g_ϕ is ControlNet function processing the image-depth input.

To maintain generation consistency, mask blending is operated at each timestep. Specifically, the latent encoding z_t at timestep t is combined with the masked region encoding $z_{M,t}$ according to masks M. The mask blending operation ensures that the content in the unmasked regions is well preserved. It can be formulated as:

$$z_t \leftarrow z_t \odot M + z_{M,t} \odot (1 - M)\,. \quad (5)$$

Updating Scheme for Inpainting. For the same area of the 3D shape, the inpainting model may generate varying appearances. We implemented an updating scheme that allows us to refine previously processed regions when more favorable viewing angles become available. Hence, masks M are divided into three distinct regions based on their visibility from the current viewpoint v_j: generate mask $M_{generate}$, keep mask M_{keep} and update mask M_{update}.

The generate masks $M_{generate}$ refer to blank areas that have never been generated before. The keep masks M_{keep} are those that have been processed before and the current viewpoint does not provide better viewing conditions. The calculation of the update mask M_{update} involves evaluating whether to refresh a region based on the similarity between its viewing directions and normals. Specifically, we define a similarity mask $M_{similarity}$ to quantify the observability of surface details from different viewing angles. For a viewpoint v_j, the similarity mask value is computed as the cosine similarity between the viewing direction d_j and the point normal N: $M_{similarity} = d_j \cdot N$. A region should be updated when the current view provides a better observation angle than any other view:

$$M_{update}^{j} = \begin{cases} 1, & \text{if } M_{similarity}^{j} > M_{similarity}^{others} \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

The final inpainting $I_{inpaint}$ is generated by combining two different denoising processes: a stronger denoising for newly generated regions (generate masks) and a weaker denoising for regions requiring updates (update masks). The final appearance is achieved as:

$$I \leftarrow I_{inpaint} \odot (1 - M_{keep}) + I \odot M_{keep}. \quad (7)$$

3.3. Gaussian Optimization

For a given viewpoint v_j, we can now generate the appearance I_j with the powerful inpainting model. The Gaussians G can be optimized through I_j. Unlike the vanilla 3DGS fitting scheme that optimizes Gaussian attributes through multiple iterations across different viewpoints, GAP operates only a single optimization pass per viewpoint, which leads to more robust Gaussian generations faithfully representing the high-quality appearance I_j. Specifically, in each view-specific optimization step, we focus exclusively on optimizing the Gaussians that represent the nearest visible surface layer from the current viewpoint, without modifying the Gaussians on the back-facing surfaces, as shown in Fig. 3. To this end, we implement a Gaussian selection scheme that identifies the first intersecting Gaussian along each viewing ray originating from pixels within the generate or update mask. To manage the computational intensity of processing numerous rays, we develop a CUDA [29] implementation that exploits GPU parallelism. accelerating the Gaussian selection process to just 3 seconds.

Surface-anchoring Mechanism. During Gaussian optimization, Gaussians that float away from their expected surface positions introduce significant challenges for multi-view inpainting and updating. These Gaussians produce incorrect occlusion relationships in subsequent viewpoints, resulting in distorted masks and further degrading the quality of generation and inpainting. To this end, we introduce a surface-anchoring mechanism in terms of a distance loss which aligns Gaussians with the zero-level set of the learned unsigned distance field. Practically, we constrain distance value at each Gaussian center, queried from f_u, to be close to zero during optimization. The distance loss is formulated as:

$$\mathcal{L}_{\text{Distance}} = \|f_u(\sigma_i)\|_2. \quad (8)$$

Scale Constraint. During optimization from a single viewpoint, some oversized Gaussians may lead to incorrect geometries which adversely affect the inpainting results of subsequent views. To address this issue, we introduce a scale loss that constrains the maximum value of s_i for each Gaussian. The *Scale Loss* is defined as:

$$\mathcal{L}_{\text{Scale}} = \left(\min(\max(s_i), \tau) - \max(s_i)\right)^2, \quad (9)$$

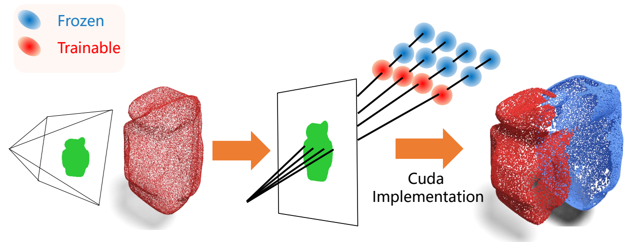

Figure 3. **Gaussian Selection scheme.** We identifies the first intersecting Gaussian along each viewing ray within generate or update masks, implemented with CUDA for efficient processing.

where τ is a predefined threshold value. The scale loss effectively prevents Gaussians from growing excessively large while still allowing sufficient flexibility to model the appearance.

Rendering Constraint. Following 3DGS [21], we also employ the *Rendering Loss* during optimization. The rendering constraint consists of an $L1$ loss term and a D-SSIM term with weights of 0.8 and 0.2 respectively:

$$\mathcal{L}_{\text{Rendering}} = 0.8L_1(I'_j, I_j) + 0.2L_{D-SSIM}(I'_j, I_j), \quad (10)$$

where I'_j is the rendered image. With the balanced weight α and β, the final optimization objective can be formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{Rendering}} + \alpha\mathcal{L}_{\text{Distance}} + \beta\mathcal{L}_{\text{Scale}}. \quad (11)$$

3.4. Diffuse-based Gaussian Inpainting

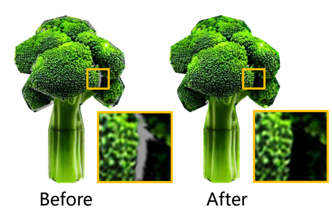

Figure 4. The Gaussian inpainting approach effectively completes the unseen regions by propagating properties from visible Gaussians.

Even with comprehensive multi-view capturing from densely sampled viewpoints, certain regions of the 3D object are still challenging to observe. As shown in Fig. 2(c), to model the appearances of the unseen areas, we propose a diffuse-based Gaussian inpainting approach. Our method effectively recovers missing appearance in the final representation, as shown in Fig. 4. Our approach operates inpainting directly in 3D space, leveraging the inherent structure and spatial relationships of the visible Gaussians. Using the Gaussian selection scheme across multiple viewpoints, we can effectively identify the unseen Gaussians $G' = \{g'_j\}_{j=1}^{M'}$, which are not optimized at any view. For the unseen Gaussians, whose positions and normal directions have already been well initialized through the Gaussian initialization scheme proposed in Sec. 3.1, we primarily focus on predicting their remaining properties, such as color, scale, and opacity.

Color Diffuse. To predict the color attributes of the unseen regions, we implement a diffusion mechanism that propagates the attributes of nearby Gaussians. For each unseen

ICCV
#2942

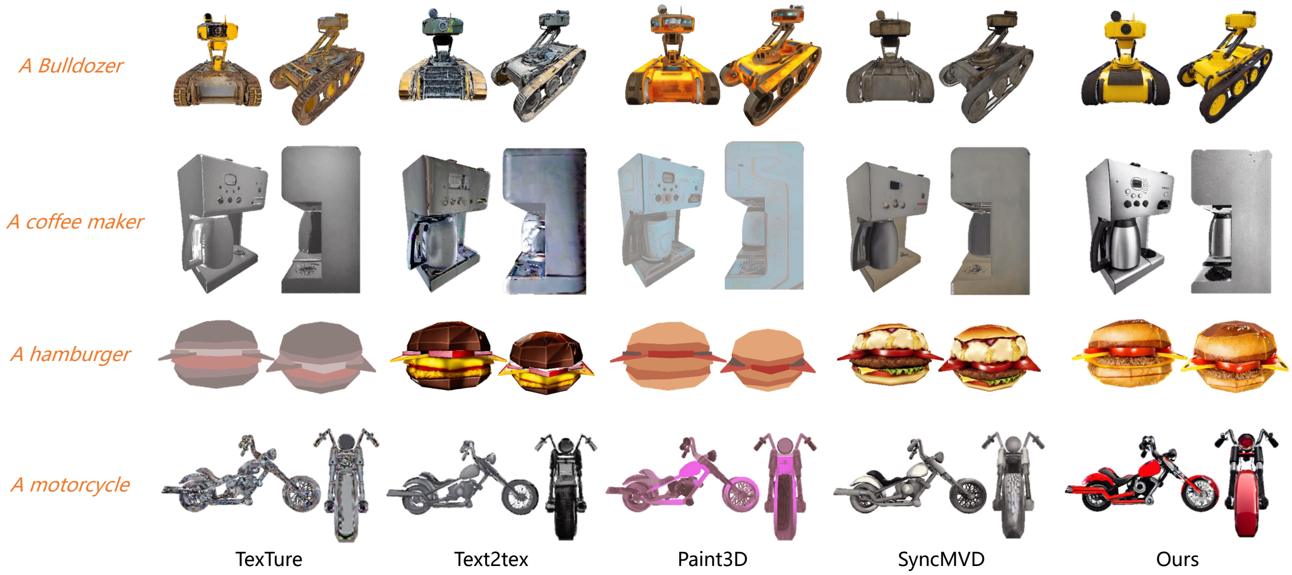

Figure 5. Visual comparison of text-guided appearance generation results on the Objaverse dataset.

Gaussian g'_j, we first locate its L nearest optimized neighbor Gaussians as the reference. We design a weighting strategy that incorporates spatial proximity, geometric consistency, and opacity reliability during color diffuse. Let o_{max} be the maximum opacity value among all neighbor Gaussians. For each valid neighbor g_i of the unseen Gaussian g'_j, we define its color weight λ_i as follows: when the angle between the normals of g_i and g'_j is less than 60 degrees, i.e., $(\mathbf{n}_i \cdot \mathbf{n}_j) > 0.5$, the weight is calculated as:

$$\lambda_i = \frac{1/d_i}{\sum_{k=1}^{L} 1/d_k} \cdot (\mathbf{n}_i \cdot \mathbf{n}_j) \cdot \frac{o_i}{o_{max}}. \tag{12}$$

Otherwise, the weight is set to 0. The distance term $1/d_i$ prevents the far Gaussians with inconsistent appearances to largely affect the color, while the normal consistency term $(\mathbf{n}_i \cdot \mathbf{n}_j)$ preserves geometric features by prioritizing color propagation between Gaussians with similar surface orientations. The opacity reliability term o_i/o_{max} ensures that Gaussians with higher opacity values have a stronger influence on the color prediction. Finally, the color c'_j of the unseen Gaussian g'_j can be formulated as:

$$c'_j = \frac{\sum_{i=1}^{L}(c_i * \lambda_i)}{\sum_{i=1}^{L} \lambda_i}. \tag{13}$$

Size Scale. To predict appropriate scales for the unseen Gaussians g'_j, we consider the L nearest neighbors (including both optimized and unseen Gaussians). The scale is adjusted based on the spatial proximity of these neighbors. The scale s'_j of an unseen Gaussian is computed as:

$$s'_j = \log(\frac{\sum_{i=1}^{L} d_i}{L}), \tag{14}$$

where d_i represents the distance between the unseen Gaussian g'_j and its neighbor g_i. We incorporate distance weighting, as larger distances indicate sparser regions that require larger scales.

Opacity Control. For predicting the opacity o'_j of an unseen Gaussian g'_j, we employ a density-based control mechanism. The opacity within a radius ρ is inversely proportional to the local Gaussian density. The opacity o'_j of an unseen Gaussian g'_j is computed as:

$$o'_j = \frac{o_0}{max(1, P/P_0)}, \tag{15}$$

where o_0 is a base opacity value, P is the number of neighboring Gaussians within a specified radius ρ, and P_0 is a reference density threshold. The opacity control scheme ensures that regions with higher Gaussian density have lower opacity values, preventing over-accumulation of color while maintaining proper surface coverage.

4. Experiments

We first evaluate GAP's core capability of text-driven appearance generation in Sec. 4.1. In Sec. 4.2, we compare GAP's performance specifically on the Point-to-Gaussian generation task with other Gaussian generation methods. Next, we further validate GAP's capability on real-world scanned point clouds, where the inputs are often incomplete in Sec. 4.3. In Sec. 4.4, we showcase GAP's scalability by applying it to scene-level point clouds. Finally, the ablation studies are shown in Sec. 4.5.

4.1. Text-Driven Appearance Generation

Datasets and Metrics. Following prior works [7, 33], we conduct experiments on the curated subset of the Objaverse [12] dataset containing 410 textured meshes across

ICCV
#2942

Figure 6. Visual comparison of point-to-Gaussian generation results on DeepFashion3D.

225 categories. Unlike previous methods that require perfect meshes, we only use a sampled point cloud of $100K$ points as input. We employ three complementary metrics: Fréchet Inception Distance (FID) [49] and Kernel Inception Distance (KID $\times 10^{-3}$) [3] for assessing image quality, and CLIP Score [32] for measuring text-image alignment. All methods use identical text prompts describing each object. We render all objects at a high resolution of 1024×1024 pixels from fixed viewpoints.

Baselines. For visual appearance, we compare GAP with state-of-the-art 3D texture generation methods: TexTure [33], Text2Tex [7], Paint3D [51], and SyncMVD [24], all of which rely on UV-mapped meshes. And the original meshes in the subset of the Objaverse dataset include artist-created UV maps. For a fair comparison with those methods under the same conditions of point cloud inputs, we reconstruct meshes from the input point clouds using the traditional Ball-Pivoting Algorithm (BPA) [2] and SOTA learning-based method CAP-UDF [54]. We then generate UV maps through xatlas [47] unwrapping.

Comparison. The quantitative comparison in Tab. 1 shows that GAP outperforms previous state-of-the-art methods. Unlike approaches relying on artist-created UV maps, GAP leverages Gaussian Splatting for inherently higher rendering quality. The performance gap is even more pronounced compared to baselines using reconstructed meshes, which suffer from topological ambiguities, connectivity errors, and geometric distortions. These issues, compounded by dense mesh reconstructions and automated UV unwrapping, often result in severe texture artifacts. In contrast, GAP bypasses UV parameterization by directly optimizing Gaussian primitives in 3D space. As shown in Fig. 5, while existing methods generate plausible appearances, they struggle with detail preservation. By directly optimizing appearance in 3D space, GAP achieves superior visual quality across object categories. A more detailed visual comparison with mesh-based methods is provided in the supplementary material.

To assess visual appearance and text alignment, we conducted a user study with 30 participants. Each participant independently evaluated results from all methods across multiple viewpoints, rating them on a scale of 1 to 5.

Table 1. Quantitative comparison with baselines on the Objaverse dataset. Best results are highlighted as 1st, 2nd and 3rd.

Method	FID↓	KID↓	CLIP↑	User Study Overall Quality↑	User Study Text Fidelity↑
TexTure [33]	42.63	7.84	26.84	2.90	3.05
Text2Tex [7]	41.62	6.45	26.73	3.48	3.62
SyncMVD [24]	40.85	5.77	27.24	3.12	3.4
Paint3D [51]	41.08	5.81	26.73	3.07	3.33
TexTure$_{BPA}$	60.69	15.98	26.62	1.46	1.62
Text2Tex$_{BPA}$	64.35	16.67	26.18	2.86	3.06
SyncMVD$_{BPA}$	60.29	14.35	26.19	2.85	3.12
Paint3D$_{BPA}$	65.36	17.37	25.14	1.45	1.45
TexTure$_{CAP}$	53.55	12.43	26.68	2.23	2.60
Text2Tex$_{CAP}$	52.78	11.09	26.78	3.03	3.57
SyncMVD$_{CAP}$	63.85	16.92	25.81	2.97	3.09
Paint3D$_{CAP}$	59.49	13.56	24.99	2.38	2.40
Ours	40.39	5.28	27.26	4.21	4.47

4.2. Point-to-Gaussian Generation

Datasets and Implementations. To evaluate GAP's effectiveness in Point-to-Gaussian generation, we conduct experiments on two datasets: the ShapeNet chair category [6] and DeepFashion3D [60]. We uniformly sample 100K points from each 3D model to generate input point clouds. GAP is compared with three state-of-the-art methods DreamGaussian [38], TriplaneGaussian [61], and DiffGS [57], all using the same 100K point clouds as input. Please refer to the supplementary for the adaptions of those baseline methods, as well as additional results.

Performance. We provide visual comparisons with baseline methods in Fig. 6, GAP consistently generates more visually appealing and geometrically accurate results compared to existing approaches. The baseline methods exhibit several key limitations. DreamGaussian, despite incorporating Score Distillation Sampling (SDS) for appearance optimization, tends to produce over-saturated appearances with unnatural colors. Additionally, its optimization process is computationally intensive and highly parameter-sensitive. TriplaneGaussian and DiffGS are fundamentally constrained by their limited-resolution triplane representa-

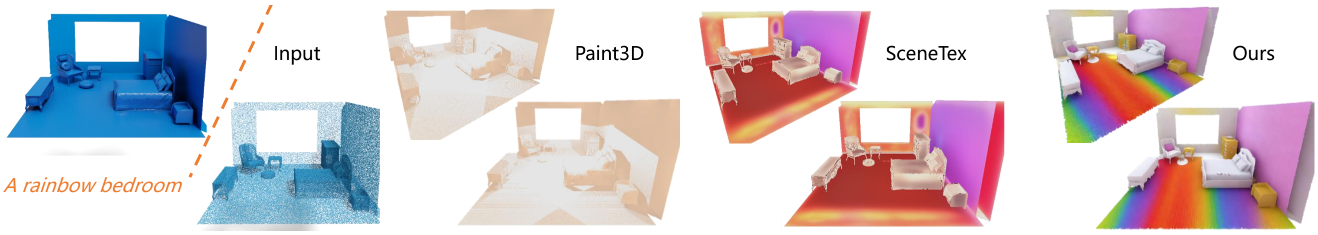

Figure 7. Scene-level Gaussianization comparison on 3D-FRONT datasets.

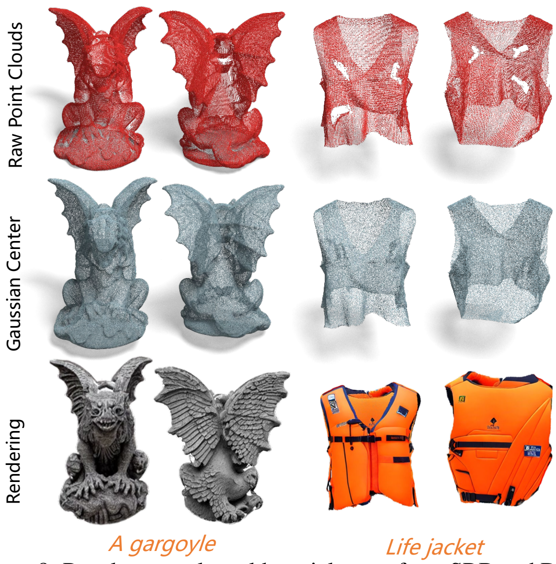

Figure 8. Results on real-world partial scans from SRB and Deep-Fashion3D datasets.

tions, limiting their ability to capture appearance details.

4.3. Gaussian Generation for Scanned Inputs

Datasets. We evaluate GAP on real-world partial scans from SRB (Scan-to-Reality Benchmark) [41] and Deep-Fashion3D [60] datasets. Both datasets contain point clouds captured by depth sensors, presenting real-world challenges such as incomplete coverage, occlusions and scanning artifacts. We directly use the raw scanned point clouds as input. **Performance.** As shown in Fig. 8, GAP successfully gaussianizes partial point clouds into complete, high-quality Gaussian representations. Our surface-anchoring mechanism effectively pull the split and cloned 3D Gaussians to fill missing regions while preserving geometric consistency. The results demonstrate that our method can robustly handle artifacts and occlusions in real-world scanned point clouds and generate visually appealing Gaussians.

4.4. Scale to Scene-Level Gaussian Generation

Datasets. We evaluate GAP on both synthetic and real-world scene datasets. For synthetic scenes, we use 3D-FRONT [14], which features diverse indoor environments. We sample 500K points from scene meshes as input. For real-world evaluation, we use raw point clouds from the 3D Scene dataset [59], which poses challenges such as complex topology, varying point densities, and scanning artifacts. **Comparision.** Compared to Paint3D [51] and Scenetex [8], our method achieves superior visual quality. As shown in Fig. 7, Paint3D fails on scene-level data, while SceneTex requires both VSD optimization [40] and additional LoRA [18] training, significantly increasing processing time. In contrast, our method produces high-quality results for complex scenes with a single optimization process. Please refer to the supplementary for more results on real-world scenes.

4.5. Ablation Study

To analyze the effectiveness of key components in GAP, we performed a series of controlled experiments. The performance was measured using three metrics: FID, KID, and CLIP Score. These metrics were computed on rendered images captured from multiple viewpoints. We evaluate some major designs of our framework in Tab. 2. Without the Scale Loss, Gaussians grow excessively large, leading to distorted results in subsequent views. The Distance Loss prevents Gaussians from drifting away from object surfaces, maintaining geometric accuracy. The diffuse-based Gaussian Inpainting ensures complete coverage in hard-to-observe regions. Each component proves essential for achieving optimal performance.

Table 2. Ablation study of key components in GAP.

Method	FID↓	KID↓	CLIP↑
Full Model	**40.39**	**5.28**	**27.26**
W/o $\mathcal{L}_{\text{Scale}}$	214.63	79.04	26.25
W/o $\mathcal{L}_{\text{Distance}}$	161.04	23.29	24.30
W/o GS Inpainting	46.37	8.77	27.21

5. Conclusion

In this paper, we presented GAP, a novel approach that generates high-quality 3D Gaussians from raw point clouds with text guidance. We design a multi-view optimization framework which learns Gaussian attributes from text-to-image diffusion models. The surface-anchoring constraint and diffuse-based Gaussian inpainting scheme are proposed to ensure geometric accuracy and appearance completion. Extensive experiments demonstrate GAP's effectiveness on both synthetic and real-world scanned data, from objects to large-scale scenes.

References

[1] Raphael Bensadoun, Yanir Kleiman, Idan Azuri, Omri Harosh, Andrea Vedaldi, Natalia Neverova, and Oran Gafni. Meta 3d texturegen: Fast and consistent texture generation for 3d objects. *arXiv preprint arXiv:2407.02430*, 2024. 2

[2] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, 1999. 7

[3] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 7

[4] Mario Botsch. Polygon mesh processing. *AK Peters*, 2010. 2

[5] Tianshi Cao, Karsten Kreis, Sanja Fidler, Nicholas Sharp, and Kangxue Yin. Texfusion: Synthesizing 3d textures with text-guided image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4169–4181, 2023. 2

[6] 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. 7

[7] Dave Zhenyu Chen, Yawar Siddiqui, Hsin-Ying Lee, Sergey Tulyakov, and Matthias Nießner. Text2tex: Text-driven texture synthesis via diffusion models. *arXiv preprint arXiv:2303.11396*, 2023. 2, 6, 7

[8] Dave Zhenyu Chen, Haoxuan Li, Hsin-Ying Lee, Sergey Tulyakov, and Matthias Nießner. Scenetex: High-quality texture synthesis for indoor scenes via diffusion priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21081–21091, 2024. 8

[9] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22246–22256, 2023. 2

[10] Wei Cheng, Juncheng Mu, Xianfang Zeng, Xin Chen, Anqi Pang, Chi Zhang, Zhibin Wang, Bin Fu, Gang Yu, Ziwei Liu, et al. Mvpaint: Synchronized multi-view diffusion for painting anything 3d. *arXiv preprint arXiv:2411.02336*, 2024. 2

[11] Julian Chibane, Gerard Pons-Moll, et al. Neural unsigned distance fields for implicit function learning. *Advances in Neural Information Processing Systems*, 33:21638–21652, 2020. 3

[12] 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. *arXiv preprint arXiv:2212.08051*, 2022. 6

[13] Robert A Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. *ACM Siggraph Computer Graphics*, 22(4): 65–74, 1988. 2

[14] Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 3d-front: 3d furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10933–10942, 2021. 8

[15] 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. 2

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2

[17] 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. 3

[18] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 8

[19] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. *arXiv preprint arXiv:2403.17888*, 2024. 3

[20] DaDong Jiang, Xianghui Yang, Zibo Zhao, Sheng Zhang, Jiaao Yu, Zeqiang Lai, Shaoxiong Yang, Chunchao Guo, Xiaobo Zhou, and Zhihui Ke. Flexitex: Enhancing texture generation with visual guidance. *arXiv preprint arXiv:2409.12431*, 2024. 2

[21] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4):1–14, 2023. 2, 3, 5

[22] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. Least squares conformal maps for automatic texture atlas generation. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 193–202. 2023. 2

[23] Jiahe Li, Jiawei Zhang, Xiao Bai, Jin Zheng, Xin Ning, Jun Zhou, and Lin Gu. Dngaussian: Optimizing sparse-view 3d gaussian radiance fields with global-local depth normalization. *arXiv preprint arXiv:2403.06912*, 2024. 3

[24] Yuxin Liu, Minshan Xie, Hanyuan Liu, and Tien-Tsin Wong. Text-guided texturing by synchronized multi-view diffusion. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024. 2, 7

[25] Longfei Lu, Huachen Gao, Tao Dai, Yaohua Zha, Zhi Hou, Junta Wu, and Shu-Tao Xia. Large point-to-gaussian model for image-to-3d generation. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 10843–10852, 2024. 2, 3

[26] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12663–12673, 2023. 2

[27] B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, and R Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, 2020. 2

[28] Mehdi Mirza. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 2

[29] NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. Cuda, release: 10.2.89, 2020. 5

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

[31] Alec Radford. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2

[32] 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. 7

[33] Elad Richardson, Gal Metzer, Yuval Alaluf, Raja Giryes, and Daniel Cohen-Or. Texture: Text-guided texturing of 3d shapes. In *ACM SIGGRAPH 2023 conference proceedings*, pages 1–11, 2023. 2, 6, 7

[34] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 3

[35] 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. 2

[36] Pedro V Sander, Steven Gortler, John Snyder, and Hugues Hoppe. Signal-specialized parameterization. 2002. 2

[37] Dewen Seng and Hongxia Wang. Realistic real-time rendering of 3d terrain scenes based on opengl. In *2009 First International Conference on Information Science and Engineering*, pages 2121–2124. IEEE, 2009. 2

[38] 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. 3, 7

[39] Jiaxiang Tang, Ruijie Lu, Xiaokang Chen, Xiang Wen, Gang Zeng, and Ziwei Liu. Intex: Interactive text-to-texture synthesis via unified depth-aware inpainting. *arXiv preprint arXiv:2403.11878*, 2024. 2

[40] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems*, 36, 2024. 8

[41] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10130–10139, 2019. 8

[42] 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. 3

[43] Xiaoyu Xiang, Liat Sless Gorelik, Yuchen Fan, Omri Armstrong, Forrest Iandola, Yilei Li, Ita Lifshitz, and Rakesh Ranjan. Make-a-texture: Fast shape-aware texture generation in 3 seconds. *arXiv preprint arXiv:2412.07766*, 2024. 2

[44] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. *arXiv preprint arXiv:2403.14621*, 2024. 3

[45] Yu-Ying Yeh, Jia-Bin Huang, Changil Kim, Lei Xiao, Thu Nguyen-Phuoc, Numair Khan, Cheng Zhang, Manmohan Chandraker, Carl S Marshall, Zhao Dong, et al. Texturedreamer: Image-guided texture synthesis through geometry-aware diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4304–4314, 2024. 2

[46] 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. 3

[47] Jonathan Young. xatlas: A Library for Mesh Parameterization. GitHub repository, 2018. 7

[48] Xin Yu, Peng Dai, Wenbo Li, Lan Ma, Zhengzhe Liu, and Xiaojuan Qi. Texture generation on 3d meshes with point-uv diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4206–4216, 2023. 2

[49] Yu Yu, Weibin Zhang, and Yun Deng. Frechet inception distance (fid) for evaluating gans. *China University of Mining Technology Beijing Graduate School*, 3, 2021. 7

[50] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19447–19456, 2024. 3

[51] Xianfang Zeng, Xin Chen, Zhongqi Qi, Wen Liu, Zibo Zhao, Zhibin Wang, Bin Fu, Yong Liu, and Gang Yu. Paint3d: Paint anything 3d with lighting-less texture diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4252–4262, 2024. 2, 7, 8

[52] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. *arXiv preprint arXiv:2404.19702*, 2024. 3

[53] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 3

[54] Junsheng Zhou, Baorui Ma, Yu-Shen Liu, Yi Fang, and Zhizhong Han. Learning consistency-aware unsigned distance functions progressively from raw point clouds. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 7

[55] Junsheng Zhou, Baorui Ma, Yu-Shen Liu, Yi Fang, and Zhizhong Han. Learning consistency-aware unsigned dis-

tance functions progressively from raw point clouds. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 3

[56] Jingqiu Zhou, Lue Fan, Xuesong Chen, Linjiang Huang, Si Liu, and Hongsheng Li. Gaussianpainter: Painting point cloud into 3d gaussians with normal guidance. *arXiv preprint arXiv:2412.17715*, 2024. 3

[57] Junsheng Zhou, Weiqi Zhang, and Yu-Shen Liu. Diffgs: Functional gaussian splatting diffusion. *arXiv preprint arXiv:2410.19657*, 2024. 2, 7

[58] Kun Zhou, Xin Huang, Xi Wang, Yiying Tong, Mathieu Desbrun, Baining Guo, and Heung-Yeung Shum. Mesh quilting for geometric texture synthesis. In *ACM SIGGRAPH 2006 Papers*, pages 690–697. 2006. 2

[59] Qian-Yi Zhou and Vladlen Koltun. Dense scene reconstruction with points of interest. *ACM Transactions on Graphics (TOG)*, 32(4):1–8, 2013. 8

[60] Heming Zhu, Yu Cao, Hang Jin, Weikai Chen, Dong Du, Zhangye Wang, Shuguang Cui, and Xiaoguang Han. Deep fashion3d: A dataset and benchmark for 3d garment reconstruction from single images. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 512–530. Springer, 2020. 7, 8

[61] 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. 3, 7