

SplatPainter: Interactive Authoring of 3D Gaussians from 2D Edits via Test-Time Training

Yang Zheng^{1,2} Hao Tan² Kai Zhang² Peng Wang²
Leonidas Guibas¹ Gordon Wetzstein¹ Wang Yifan²

¹ Stanford University
² Adobe

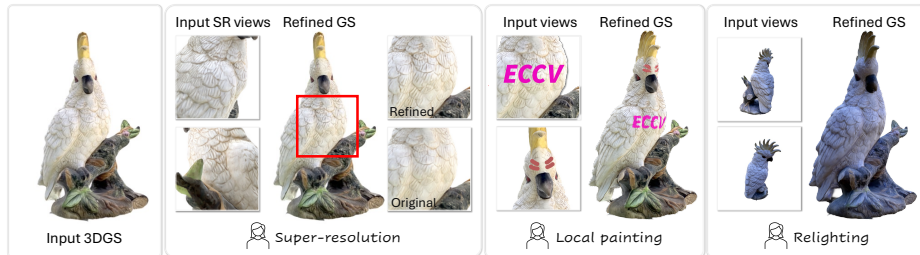


Fig. 1: We present SplatPainter, a feedforward method to support interactive and continuous authorship of 3D Gaussian assets through intuitive 2D edits, *e.g.*, high-fidelity local detail refinement, local paint-over, and consistent global recoloring, while maintaining 3D consistency and the original texture and structure details.

Abstract. The rise of 3D Gaussian Splatting has revolutionized photorealistic 3D asset creation, yet a critical gap remains for their interactive appearance refinement and editing. Existing approaches based on diffusion or optimization are ill-suited for this task, as they are often prohibitively slow, lack the precision for fine-grained control, or—most importantly—are fundamentally destructive to the original asset’s identity. To address this, we introduce SplatPainter, a state-aware feedforward model that enables continuous, high-fidelity appearance editing of 3D Gaussian assets from user-provided 2D views. Our method directly predicts updates to the attributes of a compact, feature-rich Gaussian representation and leverages Test-Time Training to create a state-aware, iterative workflow. The versatility of our approach allows a single architecture to perform diverse tasks, including high-frequency local detail refinement, local paint-over, and consistent global recoloring, all at interactive speeds while strictly preserving the asset’s original structure. Our project website is at: <https://y-zheng18.github.io/SplatPainter/>.

1 Introduction

High-quality 3D content creation is fundamental to a growing number of industries, from gaming and virtual reality to e-commerce and digital twins. Recently, 3D Gaussian Splatting (3DGS) [41] has emerged as a leading representation for

its real-time photorealistic rendering and its ability to capture complex details like hair and fur more effectively than traditional meshes. As text- and image-to-3D pipelines [45, 77] built on GS become more powerful, they enable the rapid creation of assets. However, these one-shot generation methods typically represent a “starting point” rather than a final product; their focus on global coherence often comes at the cost of local detail, creating an urgent need for efficient workflows to refine the appearance of these assets post-creation.

The critical missing piece in the modern 3D workflow is a framework for interactively and intuitively editing the appearance of existing 3DGS assets. For content creation to be truly powerful, users need to make precise adjustments with familiar 2D tools, applying edits to arbitrary views—including close-ups for fine-grained refinement. This is a non-trivial challenge that requires addressing three core difficulties: (1) achieving precise, controllable updates that faithfully follow user intent; (2) preserving the original identity and intricate structural details of unedited regions; and (3) operating at interactive speeds. As illustrated in Fig. 2, maintaining identity is not a “given” even for texture-only edits. Existing tools often introduce significant and unintended identity drift or fail to match the user’s specific 2D intent, leading to results that are either perceptually inconsistent with the original asset.

Current 3D editing approaches are fundamentally ill-suited for this task of interactive refinement. Many methods rely on an intermediate step of generating multiview images to match a 2D edit before lifting them to 3D via reconstruction [3, 4, 7, 11, 17, 26, 44, 56, 83]. This process is prone to 3D inconsistencies that destroy original textural details and lead to “blurry” results that fail to capture the user’s specific intent. Alternatively, methods that employ diffusion directly on structured 3D latents [2, 16] bypass multiview generation but suffer from “spatial edit leakage”. In these cases, appearance changes frequently bleed into unintended regions – a limitation common to latent diffusion models [1]. Regardless of the underlying paradigm, these methods share a prohibitive latency. Whether based on iterative diffusion sampling or per-scene optimization, they are far too slow for real-time feedback, often requiring hundreds of seconds for a single change and breaking the fluid creative process required by artists.

In this work, we introduce SplatPainter, a state-aware feedforward framework designed for fast and identity-preserving appearance editing of 3D Gaussian assets. By resolving 2D-to-3D consistency directly in the Gaussian representation, we bypass the need to finetune or repurpose specific generative models. Instead, our approach acts as a versatile appearance-lifting module that can incorporate 2D edits from diverse sources—including manual user paint-overs, super-resolution models, or advanced generative editors like Nano Banana [21] – without requiring the destructive multiview regeneration typical of existing workflows. Our model directly predicts updates to Gaussian attributes (e.g., color and opacity) from these image-space edits, ensuring the asset’s original structural integrity remains intact. To facilitate a continuous creative process, we utilize Test-Time Training (TTT) [68, 80, 100] to make the model state-aware. This allows the network to adapt to user inputs on the fly, integrating new ap-



Fig. 2: Challenges in Controllability and Identity Preservation. Existing methods struggle to propagate precise 2D edits into 3D space. Approaches that rely on multi-view generation (e.g., DGE [8]) struggle to match user intention (e.g., the drop of tear) and often result in 3D inconsistencies and blurry textures. Even direct 3D methods (e.g., VoxHammer [2], VineDresser3D [16]) can introduce global identity drift during localized updates (e.g., color shift and detail distortion). In contrast, SplatPainter provides precise, intent-matching updates—leveraging external editors like Nano Banana for high-quality guidance—while strictly maintaining the original asset’s identity.

pearance information while maintaining the underlying identity of the original 3D asset.

Our main contributions can be summarized as follows:

1. **A Novel Feedforward Editing Framework:** We introduce SplatPainter, a state-aware feedforward model for the interactive and continuous editing of 3DGS. By leveraging TTT, our method iteratively incorporates user edits from 2D views, directly updating the attributes of a compact and feature-rich Gaussian representation without destructive regeneration.
2. **Versatile and Intuitive Applications:** We evaluate our framework on two novel and practical applications: high-fidelity local detail refinement and consistent global relighting. The unified approach provides users with precise control over both fine-grained texture details and broad appearance changes from 2D edit(s).
3. **State-of-the-Art Performance at Interactive Speeds:** Through extensive experiments, we show that SplatPainter significantly outperforms existing methods in both quality and speed. Our approach achieves superior visual results while operating at interactive rates, making it a truly practical solution for unlocking fluid, real-time creative workflows in 3D content authoring.

2 Related Work

Our work lies at the intersection of editing for modern 3D representations like NeRFs [50] and 3D Gaussian Splatting (3DGS) [41], and continuous 3D reconstruction. We begin by reviewing the evolving ecosystem of direct GS editing before discussing broader paradigms for image-driven 3D editing and the architectural principles that enable our iterative approach.

The Evolving 3D Gaussian Splatting Ecosystem. The wide adoption of 3DGS [41] and its plethora of extensions [27] has propelled research into making GS assets directly editable, leading to the emergence of interactive tools for

manual selection and editing [36, 47, 48, 54]. More advanced methods achieve real-time geometric deformation through auxiliary cage structures [35, 70, 90] or use feed-forward networks for tasks like splat densification [15, 52]. Other works augment the GS representation itself to enable new modalities like physics simulation [91]. While these approaches showcase a clear demand for direct and efficient GS manipulation, they primarily focus on geometry modification. They do not provide a mechanism for our task: intuitive, fine-grained appearance editing guided by 2D image inputs.

Image-Driven 3D Editing. Image-driven 3D editing is a central challenge, with three dominant strategies emerging. The first, an optimization-heavy approach, updates a 3D representation by optimizing it to match a set of edited 2D views. This paradigm is used for tasks like geometry and texture refinement [14, 58], super-resolution [28, 43, 64, 74, 84, 85, 89, 94, 95], and CLIP-guided stylization [12, 30, 33, 42, 49, 57, 65, 75, 76, 78, 104]. Score Distillation Sampling [55] is a popular variant, where optimization is guided by text prompts instead of explicit images [9, 20, 23, 24, 38, 46, 53, 62, 73, 92, 97, 106, 107]. Although these methods can produce high-quality results, their reliance on slow, per-scene optimization makes them unsuitable for interactive workflows. The second strategy, generative reconstruction, uses a multi-view diffusion model to generate consistent 2D edits from a prompt [3, 4, 7, 8, 11, 17, 26, 32, 44, 56, 71, 82, 83], which are then lifted to a new 3D asset using a feedforward reconstruction model [34, 93, 98]. This workflow, while faster than full optimization, is still too slow for real-time use and inherently destructive as it replaces the original asset. Finally, recent approaches explore direct 3D editing via structured diffusion latents to bypass multiview generation [2, 16, 86, 88]. While methods enable more coherent updates by operating in a native 3D latent space, they often suffer from spatial leakage where edits bleed into unedited regions due to the global receptive fields of diffusion priors. Our method, in contrast to these strategies, is fast, non-destructive, and handles fine-grained image edits with strict locality.

Continuous and State-Aware Architectures. Our architecture is inspired by continuous reconstruction, where a 3D representation is progressively improved with new observations. This concept spans classical SLAM [25, 29, 51, 69, 105] and Structure-from-Motion [1, 60] to modern learning-based methods using recurrent or memory-augmented networks to maintain state [18, 31, 37, 39, 61, 66, 102]. Recent frameworks explicitly maintain latent states for continuous updates [10, 79, 81], while in the 3DGS domain, this is often done via continual optimization [96]. Drawing from these state-aware designs, we use test-time training (TTT) [100] to enable our feedforward model to iteratively incorporate user edits, achieving an efficient, online refinement workflow.

3 Method

Given a 3DGS asset $\mathcal{G} = \{g_i\}_{i=1}^N$, our goal is to efficiently update Gaussian attributes—position, scale, opacity, rotation, and spherical harmonics—based on user-provided image-space edits such as zoom-in super-resolution, graffiti-style

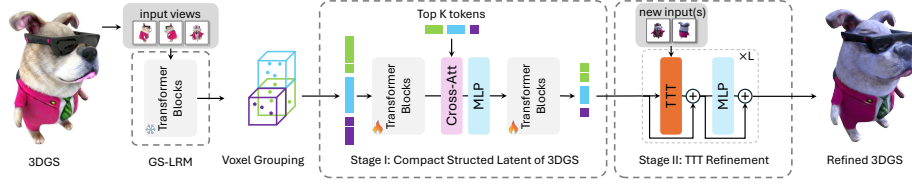


Fig. 3: Overview. Given an input 3DGS asset, the framework first performs a one-time preprocessing step. The asset is rendered from multiple views to generate feature-rich inputs from a Gaussian LRM. Stage I compresses this into a compact latent representation via a local transformer. The interactive editing loop in Stage II then iteratively refines this latent representation using new 2D user edits (new input(s)) and a Test-Time Training (TTT) module to produce the final edited 3DGS.

markings, or global recoloring (see Fig. 1). Our key contribution is a feed-forward, state-aware pipeline composed of two key components: (1) a compact and feature-rich Gaussian representation extracted via a one-time preprocessing step, and (2) a refinement module with Test-Time Training (TTT) layers [100] that incorporate new user inputs to iteratively update the scene (Fig. 3).

3.1 Compact and Feature-Rich 3DGS

Intelligently propagating a 2D edit requires a 3D representation with rich global context. Our strategy is to leverage a Gaussian large reconstruction model (LRM) [13, 93, 98] to create a high-fidelity, identity-preserving copy of the asset that is inherently enriched with such features. To accomplish this, we first render the input 3DGS from a set of canonical viewpoints. These rendered images serve as the input for the LRM, which then processes them to generate a new, dense cloud of Gaussians where each point is associated with a rich feature vector reflecting the global context. While this provides the necessary context, the LRM’s native output is a dense, unstructured cloud of Gaussians—unsuitable for interactive editing due to high redundancy and memory overhead. Therefore, we introduce a one-time preprocessing step: a voxel-based transformer that compresses and regularizes this dense output. This process extracts the globally-informed features into a compact, structured representation ideal for efficient, iterative refinement.

Preliminary: Gaussian LRM. Recent Gaussian LRMs provide a strong foundation for generating high-quality Gaussian Splatting assets from sparse multi-view images. A common approach in these frameworks is to process input multi-view images $\{I_v\}_{v=1}^V$ with a transformer-based backbone. This backbone typically extract image tokens from patchified multiview images and then feed the tokens through a chain of self-attention transformer blocks to obtain feature tokens $\{F_v\}_{v=1}^n$. These tokens are then decoded to predict per-pixel Gaussian attributes g_i . Our work builds on this architectural paradigm to acquire the initial feature-rich Gaussians before our compression stage.

Voxelization. To impose a 3D structure, we first reorganize the reconstructed Gaussians into a regular latent grid. We voxelize the scene, grouping nearby

Gaussians into local cells. Each resulting voxel $\mathcal{V} = \{(f_k, g_k)\}_{k=1}^{N_k}$ contains N_k Gaussians, where each Gaussian is associated with its feature token f_k from the LRM backbone and its Gaussians attributes g_k .

Local Voxel Transformer. On top of this structured representation, we introduce a local voxel transformer to produce compact and context-aware latent representations for Gaussians in each voxel. This is a two-stage process of aggregation and distillation. Given a voxel $\mathcal{V} = \{(f_k, g_k)\}_{k=1}^{N_k}$, we first aggregate intra-voxel relations via a transformer encoder consisting of L_{enc} layers of self-attention [72]:

$$\mathcal{Z} = \text{TransformerEncoder}(\mathcal{V}), \quad (1)$$

where $\mathcal{Z} = \{z_k\}_{k=1}^{N_k}$ is the set of context-aware latent codes for each Gaussian within the voxel. To reduce redundancy and summarize this information, we select the top- K most significant Gaussians based on their opacity α and use their latent codes as queries in a cross-attention mechanism:

$$\tilde{\mathcal{Z}} = \text{CrossAttn}(\{z_i\}_{i \in \text{TopK}_\alpha}, \mathcal{Z}). \quad (2)$$

This step effectively distills the information from all Gaussians in the voxel into a compact set of features. Finally, a transformer decoder with L_{dec} self-attention layers refines these compressed features into the final, compact voxel latent:

$$\tilde{\mathcal{V}} = \text{TransformerDecoder}(\tilde{\mathcal{Z}}). \quad (3)$$

The resulting of voxel latents $\{\tilde{\mathcal{V}}\}$ provide a structured and memory-efficient representation ready for downstream refinement. In practice, we find $K = \max(\lfloor 0.25N_k \rfloor, 1)$, *i.e.* keeping up to 25% Gaussians per voxel provides sufficient compression without inducing quality degradation.

3.2 Refinement via Test-Time Training (TTT)

To enable iterative editing, our refinement module must adapt to new user inputs on the fly without overwriting the knowledge of the original asset. We achieve this using Test-Time Training (TTT) [5, 6, 40, 67, 68, 80, 100], which updates a small set of “fast weights” [59] while keeping the main network backbone frozen. This allows our model to become state-aware, integrating new information from a user’s edit before applying it to the 3D scene.

Preliminary: Test-Time Training. Test-Time Training adapts a model to a new test sample by defining a self-supervised task at inference time. Within a transformer layer, input tokens are projected into queries (q), keys (k), and values (v) by linear layers with “slow weights”. TTT introduces a small, adaptable mapping network, f_W , governed by a set of “fast weights” W . For a given test sample, these fast weights are updated via gradient descent on a self-supervised loss:

$$\mathcal{L}_{\text{adapt}} = \mathbb{E}_{(k,v) \sim \text{TestSample}} [\ell_2(f_W(k), v)], \quad (4)$$

where the goal is for f_W to learn to reconstruct the value from the key for pairs drawn from the test sample. Once adapted, the module processes queries from the

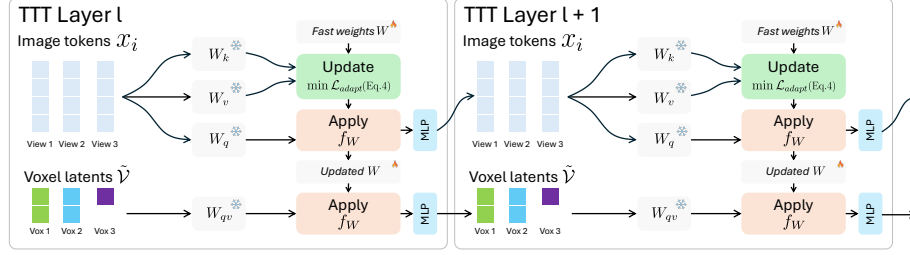


Fig. 4: TTT operations. Fast weights W are iteratively updated using new input views and subsequently applied to the voxel GS latents after seeing all the new inputs. During training, all the weights are optimized. During inference, the slow weights are fixed and only fast weights are updated.

same sample to produce the final output $o = f_W(q)$. This mechanism separates stable, general knowledge (slow weights) from instance-specific adaptation (fast weights).

Gaussian Refinement via TTT. Our refinement module is a deep transformer composed of a series of blocks, where each block contains a TTT layer followed by a feedforward network (MLP), as illustrated in Fig. 3. Details of the network architecture are shown in Fig. 4. Within each TTT layer, the model first adapts to the user’s edit and then applies this adaptation in parallel to both the image and the voxel latents.

First, the fast weights W of the mapping network f_W are updated. The set of image tokens from all new user views, $X_{img} = \{x_i\}$, are projected into key and value vectors using frozen (slow) projection heads, W_k and W_v :

$$k_i = W_k x_i, \quad v_i = W_v x_i, \quad \text{for } x_i \in X_{img}. \quad (5)$$

The fast weights W are updated by minimizing Eq. (4) over all these (k_i, v_i) pairs. This single update step infuses the TTT layer with the information from the entire user edit.

Once the fast weights are updated, the adapted mapping network f_W is applied to two parallel processing streams. For the image stream, the image tokens are projected into queries using a slow head W_q , and the adapted network produces updated image tokens:

$$x_i \leftarrow f_W(W_q x_i). \quad (6)$$

For the 3D scene stream, the voxel latents $\tilde{V}$ are projected into queries using a separate slow head W_{qv} , and the same adapted network produces the updated voxel latents:

$$\tilde{V} \leftarrow f_W(W_{qv} \tilde{V}). \quad (7)$$

The outputs of both streams are then passed through their respective MLPs. The entire block is repeated multiple times, allowing the model to progressively refine the 3D asset by repeatedly conditioning on the user’s input. Following

recent work [100], we implement the parametric mapping f_W as a SwiGLU-MLP [63].

3.3 Applications

The unified design of our framework supports a wide range of editing tasks:

- **Local Edits.** Our method enables refinement of high-frequency details from sparse close-up views (super-resolution) and integration of user-specified patches such as graffiti or text edits (see Fig. 1). These edits are applied to the visible voxels corresponding to the edited input views. When a new view is provided, we perform ray tracing to identify voxels that first intersect with the camera rays, ensuring updates are localized to intended region and the rest of the details are naturally preserved. In the local region, we further update the original GS tokens using refined tokens by adding an extra TTT layer, then merge the two token sets—effectively doubling the number of Gaussians to introduce additional fine-grained structures. Please refer to the supplementary material for additional implementation details.
- **Global Edits.** In addition to local refinement, our network also supports consistent global modifications. By applying the updated fast weights from TTT layers to all Gaussian tokens, our method enables coherent changes to appearance attributes across the entire scene. This design allows users to perform holistic edits while preserving geometry and fine structures of the original asset. We demonstrate relighting as a representative example.

Together, these components constitute SplatPainter: a feedforward, state-aware framework for interactive 3DGS refinement. Both local and global editing modules are trained with an image-based reconstruction loss:

$$\mathcal{L}_{\text{recon}} = \|I_{\text{render}} - I_{\text{gt}}\|_2 + \lambda_{\text{perc}} \|\phi(I_{\text{render}}) - \phi(I_{\text{gt}})\|_2, \quad (8)$$

where $I_{\text{render}} = R(\text{MLP}(\tilde{\mathcal{V}}), P_v)$ denotes the rendered image from the refined Gaussians decoded from $\tilde{\mathcal{V}}$ (note here we decode all the GS attributes, i.e., positions, SHs, opacity, rotation such that both geometry and appearance are updated) under camera pose P_v , and $\phi(\cdot)$ is the LPIPS feature extractor [99].

4 Experiment

In this section, we explain our experimental setup, baselines, and evaluation results across our three primary applications: local refinement, creative editing, and global relighting.

4.1 Setup

Data. We use Objaverse1.0 [22] and partial TexVerse [103] for training. For local editing tasks, we render 32 2K-resolution images and simulate user inputs:

zoomed-in crops for refinement, and overlay strokes/graffiti for creative editing. For relighting, we render $32 \times 512 \times 512$ images under 24 lighting conditions using randomly sampled HDRI maps. Evaluation is performed on 370 held-out assets from TexVerse, selected for their high-frequency texture details. The evaluation data consists of 370 examples from TexVerse that are not used for training filtered by auto-captioning, focusing on assets with high-resolution textures and rich high-frequency details to better showcase our model’s capability in fine-grained refinement.

Baselines. We compare against state-of-the-art methods tailored to each task.

- *For Local Refinement:* We compare against direct reconstruction (GS-LRM [98]), feedforward densification (GenDen [52]), and optimization approaches including 3DGS [41] and SRGS [28] specialized for 3DGS refinement.
- *For Creative Editing:* We compare against VoxHammer [2], a recent method for editing via structured 3D latents.
- *For Global Relighting:* We compare against ReLitLRM [101], a generative relighting model and direct 3DGS optimization.

Note on Multiview Diffusion: Ideally, we would compare against image-conditioned multiview diffusion methods (the “Edit-Generate-Lift” paradigm). However, current state-of-the-art methods in this category are either strictly text-conditioned [8, 32] or have not released code (e.g., CMD [44]). To illustrate the limitations of this paradigm, we provide a qualitative comparison with DGE [8] in Fig. 2, but we exclude it from quantitative tables due to the input modality mismatch.

Implementation Details. In Stage I, the Gaussian LRM is based on GS-LRM [98], extended to produce view-dependent Gaussians with 4 orders of SH. The local transformer contains six encoder self-attention blocks, one cross-attention block, and six decoder self-attention blocks (see Fig. 3), each using 512 dimensions and 8 heads implemented with Flash-Attention [19]. In Stage II, the refinement network has 8 TTT layers. The fast weights are obtained by taking 5 gradient descent steps using the Muon-update rule following Zhang et al. [100]. The updated voxel latents $\hat{\mathcal{V}}$ are decoded into Gaussian attributes through one layer MLP. We first train the local transformer in stage I, then freeze it to train the refinement network. Both stages use image reconstruction loss (Eq. (8)) with $\lambda_{perc} = 0.5$. Stage I uses 9 to 16 input images; Stage 2 trains with 1 to 4 new inputs. All inputs are 512×512 , downsampled from renderings, except for refinement views which are cropped from 2K images. Training uses 16 A100 80GB GPUs; testing is reported on a single A100.

4.2 Evaluation on Local Editing

We evaluate two distinct local editing scenarios: *High-Fidelity Refinement*, where the goal is to recover missing details from zoomed-in views, and *Interactive Creative Editing*, where the goal is to apply artistic changes (paint-overs, generative edits) while preserving the asset’s identity.

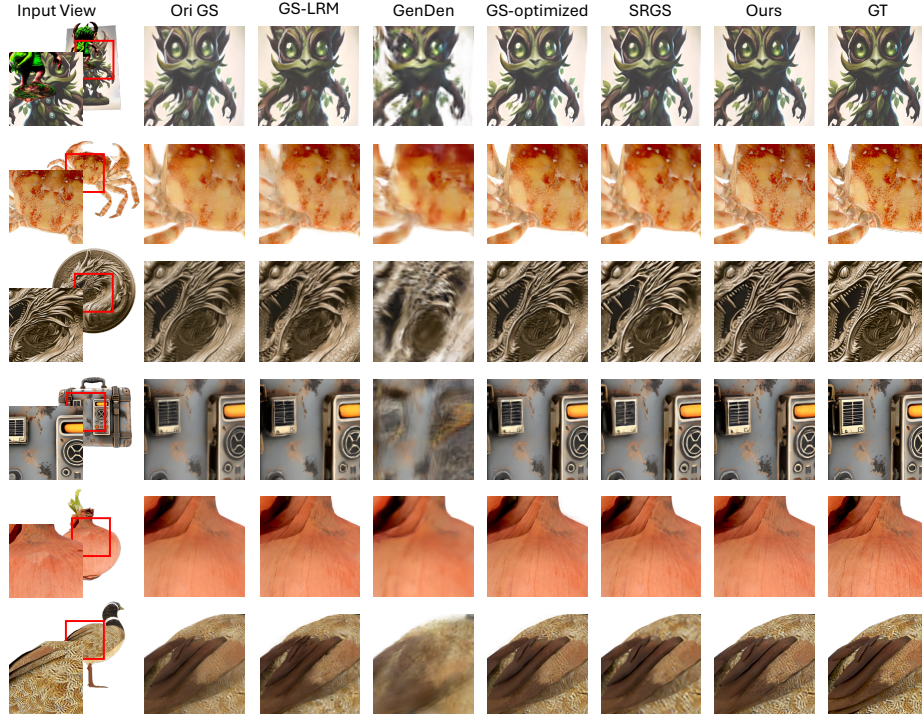


Fig. 5: Qualitative evaluation for local refinement. Given a zoomed-in input view (left), we compare the refined results between baseline methods and ours from a novel view close to the zoomed-in view. Our method recovers the details provided by the input zoomed-in views, and produces much sharper features compared to direct reconstruction (GS-LRM [98]) or upsampling method (GenDen [52]), and is on-par or better than optimization-based approaches but using a fraction of the time (see Tab. 1).

High-Fidelity Refinement In this task, a user supplies high-resolution 2D zoom-in views to refine specific regions of the 3D asset. We use 9 base views for initialization and 2 additional zoomed-in views for refinement.

Results. We evaluate image quality on novel views sampled near the edit region. As shown in Tab. 1 and Fig. 5, our method achieves the best visual and quantitative performance. Compared to feedforward methods (GS-LRM [98], GenDen [52]), our method recovers significantly sharper high-frequency textures. While optimization-based methods ([28, 41]) can achieve good sharpness, they require hundreds of seconds per update. Our method achieves superior visual quality at interactive speeds (~ 0.3 s per edit after initial setup), enabling fluid online applications.

Interactive Creative Editing This task evaluates the model’s ability to incorporate artistic edits—such as graffiti, manual paint-overs, or generative modifications via tools like Nano Banana [21]—into the 3D asset. The core challenge

Table 1: Quantitative evaluation for local refinement. SplatPainter achieves the best performance on quality and efficiency, enabling interactive online refinement while preserving high-fidelity details.

Method	PSNR ($\uparrow$)	SSIM ($\uparrow$)	LPIPS ($\downarrow$)	Time (s)
GS-LRM [98]	20.28	0.5185	0.3873	0.92
GenDen [52]	17.32	0.5378	0.5979	1.68
3DGS [41]	20.52	0.6553	0.3399	323
SRGS [28]	21.01	0.6759	0.3540	495
Ours (<i>stage I</i>)	20.84	0.5698	0.3782	1.54
Ours (<i>stage II</i>)	21.07	0.5780	0.3673	1.54+0.64

Table 3: Quantitative evaluation for creative editing. We measure photometric metrics on edited regions (compared to input edits) and non-edited regions before and after editing.

Method	Edited Region			Non-edited Region		
	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
VoxHammer [2]	11.25	0.2165	0.5036	13.86	0.4893	0.2906
VineDresser3D [16]	11.99	0.6092	0.3506	15.79	0.7437	0.2678
Ours	25.61	0.9380	0.0471	33.02	0.9560	0.0310

Table 2: Quantitative results for global relighting. SplatPainter achieves the best reconstruction accuracy while maintaining interactive speed. Performance improves steadily as more relit views are provided, demonstrating robust global propagation of lighting changes.

Method	PSNR ($\uparrow$)	SSIM ($\uparrow$)	LPIPS ($\downarrow$)	Time (s)
RelitLRM [101]	21.68	0.8726	0.0912	22.1
3DGS [41]	21.32	0.8613	0.1225	320
GS-LRM [98]+CrossAtt	24.05	0.8970	0.0802	1.54+0.13
Ours (<i>1 view</i>)	24.56	0.8986	0.0721	1.54+0.08
Ours (<i>2 views</i>)	24.71	0.8991	0.0719	1.54+0.11
Ours (<i>4 views</i>)	25.03	0.9011	0.0712	1.54+0.20

Table 4: Evaluation of our voxel compression network. Our LocalAtt + feats query design achieves the highest reconstruction accuracy with minimal quality loss.

Method	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
VoxelAtt	30.65	0.9193	0.0804
LocalAtt + latent query	33.84	0.9349	0.0264
LocalAtt + feats query	34.02	0.9387	0.0262
Pre-Compression (GS-LRM)	34.19	0.9283	0.0249

here is *Identity Preservation*: applying the edit without distorting the unedited geometry or appearance.

Comparisons. We compare our approach with VoxHammer [2] and VineDresser3D [16]. For fair comparison, we use the same 2D edited image (generated by Nano Banana) as input for both methods. As illustrated in Fig. 2 and Fig. 6, VoxHammer and VineDresser3D often struggle with “spatial leakage”, where local edits cause unintended shifts in global geometry or color tone (e.g., the robot’s eye shape or facial structure). In contrast, SplatPainter’s voxel-based TTT update is performed locally, ensuring the edit is applied faithfully while the rest of the asset remains bit-exact to the original. To quantify this, we measure the photometric losses of the edited and unedited regions of the assets (Tab. 3) on 10 generated examples from Trellis [87]. Our method achieves great identity preservation, confirming that our voxel-based updates prevent the identity drift common in latent diffusion approaches. For multi-view generation methods, we choose a representative method, DGE [8], and show qualitative results in Fig. 7, where DGE fails to interpret the fine-grained user editing intent and generates artifacts that undermine the original structure.

4.3 Evaluation on Global Editing

Finally, we evaluate global refinement through relighting. The user provides one or more “relit” views (simulating a lighting change), and the model must propagate this new illumination globally.



Fig. 6: Comparison of creative editing results. 3DGS assets are generated from Trellis [87]. Here we show our results and VoxHammer [2], and VineDresser3D [16].

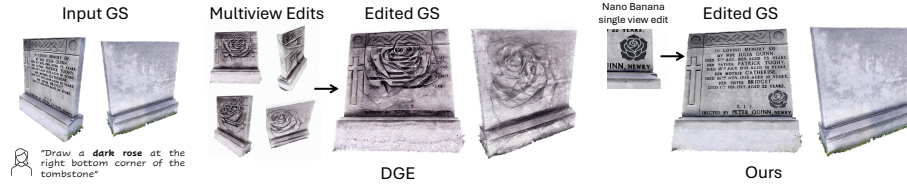


Fig. 7: Comparison of creative editing results between DGE [8] and ours.

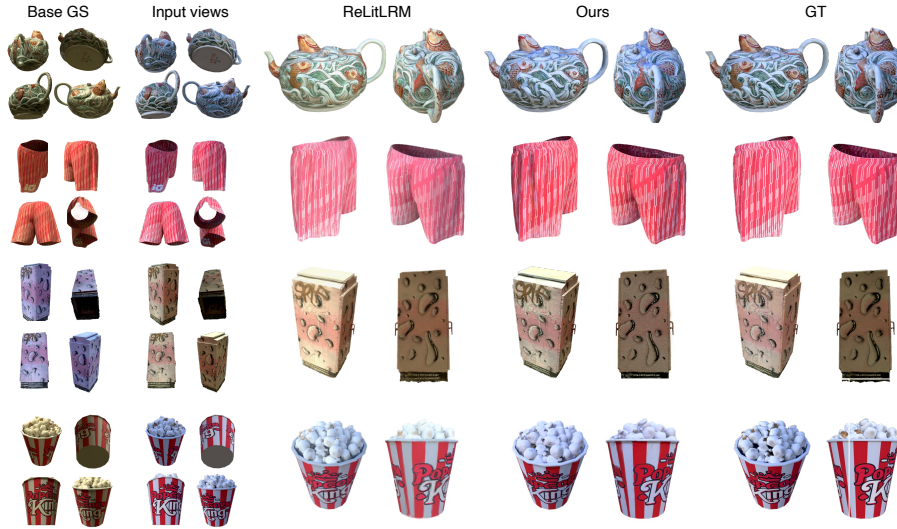


Fig. 8: Global relighting comparison. ReLitLRM takes an environment map as input to synthesize new lighting, while our method leverages a few relit views as direct input. Though not strictly relighting, our method offers a practical path for appearance transfer, achieving consistent shadows, accurate color propagation, and closer alignment with ground truth.

Results. Tab. 2 compares our method against ReLitLRM [101]. Our approach consistently achieves higher reconstruction accuracy. Notably, ReLitLRM often suffers from color tone mismatches (see Fig. 8), whereas our TTT-based propagation ensures consistent shading across views. Furthermore, our performance



Fig. 9: The two-stage pipeline on the local refinement task. Stage I achieves significant compression while maintaining visual fidelity. Stage II refines these compact GS, improving detail and overall quality based on the user’s input.

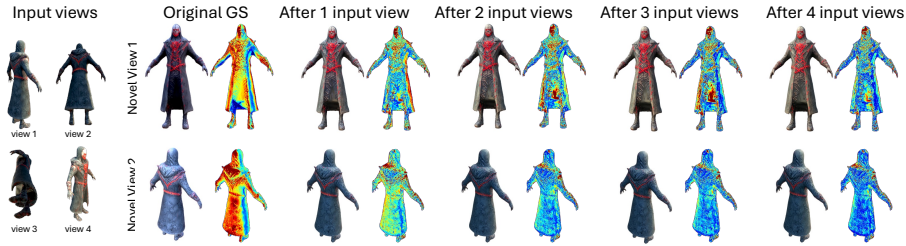


Fig. 10: Iterative updating on a global relighting task. From left to right: original GS, refined GS with 1-4 input views, where we show the rendered images and error maps. Our model picks up partial shading hint (see the first input view) and propagates it coherently to the unseen part.

scales with user input: adding more views steadily improves fidelity (Fig. 10), demonstrating the robust adaptability of our state-aware architecture.

4.4 Analysis

GS-Voxel Compression. The goal of Stage I is to create a compact Gaussian representation while maintaining the visual fidelity necessary for editing. In Tab. 4, we quantify this by comparing different voxelization strategies against the uncompressed *Pre-Compression (GS-LRM)* baseline. Our chosen design (*LocalAtt + feats query*) achieves the highest accuracy. Alternative designs like *VoxelAtt* (averaging tokens) or *LocalAtt + latent query* (generic query) result in significant information loss, as evidenced by the lower PSNR and SSIM scores. To verify the effectiveness of our two-stage approach, Fig. 9 illustrates the visual quality and Gaussian count at each step. Stage I achieves aggressive compression—reducing the number of Gaussians by approximately while maintaining high visual fidelity. Stage II then builds upon this efficient representation, refining the high-frequency details to surpass the initial quality.

TTT Update. As shown in Fig. 10, our refinement quality improves with additional views. We compare our TTT refinement against a cross-attention baseline (replacing TTT layers with cross-attention). The variant achieves lower PSNR (Tab. 2) and could introduce visual artifacts (Fig. 11). In contrast, our TTT design leads to more coherent global propagation.

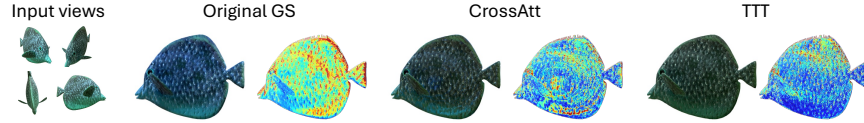


Fig. 11: Comparison between cross attention update and TTT update. Our TTT design leads to more coherent relighting results.



Fig. 12: More results. We show our refinement results on generated 3DGS assets from Trellis [87] given local super-resolution and editing (graffiti, recoloring) inputs (better viewed zoomed in).

Generalization. We further test our model on generated 3DGS from Trellis [87]. As shown in Fig. 12 and Fig. 6, our method generalizes well to generated assets, refining geometry-aligned textures from both super-resolution inputs and creative edits.

Conflict Inputs. We further evaluate our method under conflicting edit inputs. As shown in Fig. 13, different input views may introduce inconsistent edits to the same region. Our TTT refinement resolves these conflicts by updating the 3DGS representation according to the most recent input observation. As additional views arrive, the model progressively replaces outdated edits, producing a user desired final result.



Fig. 13: Results of our method when handling conflict inputs. Our model resolves conflict inputs by refining 3DGS based on the latest input information.

5 Conclusion and Future Work

We presented SplatPainter, a feedforward, state-aware framework for interactive editing and refinement of existing 3D Gaussian Splatting (3DGS) assets. Our method introduces a compact voxel-based representation that substantially reduces redundancy while maintaining visual fidelity, and a test-time training (TTT) refinement module that efficiently incorporates new user inputs to achieve

high-quality updates. Together, these designs enable both local and global editing tasks such as super-resolution, graffiti, and relighting, in an interactive manner.

While our current design focuses on appearance updates and does not introduce new geometry, future work could extend the voxelization scheme to a multiresolution representation and incorporate geometric editing capabilities for more comprehensive scene manipulation. With suitable geometry-editing datasets, the proposed framework could be directly trained to support large-scale structural updates, which we leave for future exploration.

References

1. Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S.M., Szeliski, R.: Building rome in a day. *Communications of the ACM* **54**(10), 105–112 (2011)
2. et al., L.: Voxhammer: Training-free precise and coherent 3d editing in native 3d space. *3DV* (2026)
3. Bar-On, R., Cohen-Bar, D., Cohen-Or, D.: Editp23: 3d editing via propagation of image prompts to multi-view. *arXiv preprint arXiv:2506.20652* (2025)
4. Barda, A., Gadelha, M., Kim, V.G., Aigerman, N., Bermano, A.H., Groueix, T.: Instant3dit: Multiview inpainting for fast editing of 3d objects. In: *CVPR*. pp. 16273–16282 (2025)
5. Behrouz, A., Razaviyayn, M., Zhong, P., Mirrokni, V.: It’s all connected: A journey through test-time memorization, attentional bias, retention, and online optimization. *arXiv preprint arXiv:2504.13173* (2025)
6. Behrouz, A., Zhong, P., Mirrokni, V.: Titans: Learning to memorize at test time. *arXiv preprint arXiv:2501.00663* (2024)
7. Chen, H., Shen, B., Liu, Y., Shi, R., Zhou, L., Lin, C.Z., Gu, J., Su, H., Wetzstein, G., Guibas, L.: 3d-adapter: Geometry-consistent multi-view diffusion for high-quality 3d generation. *arXiv preprint arXiv:2410.18974* (2024)
8. Chen, M., Laina, I., Vedaldi, A.: Dge: Direct gaussian 3d editing by consistent multi-view editing. *arXiv preprint arXiv:2404.18929* (2024)
9. Chen, M., Xie, J., Laina, I., Vedaldi, A.: Shap-editor: Instruction-guided latent 3d editing in seconds. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 26456–26466 (2024)
10. Chen, X., Chen, Y., Xiu, Y., Geiger, A., Chen, A.: Ttt3r: 3d reconstruction as test-time training. *arXiv preprint arXiv:2509.26645* (2025)
11. Chen, Y., Chen, Z., Zhang, C., Wang, F., Yang, X., Wang, Y., Cai, Z., Yang, L., Liu, H., Lin, G.: Gaussianeditor: Swift and controllable 3d editing with gaussian splatting. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 21476–21485 (2024)
12. Chen, Y., Chen, R., Lei, J., Zhang, Y., Jia, K.: Tango: Text-driven photorealistic and robust 3d stylization via lighting decomposition. *Advances in Neural Information Processing Systems* **35**, 30923–30936 (2022)
13. Chen, Y., Xu, H., Zheng, C., Zhuang, B., Pollefeys, M., Geiger, A., Cham, T.J., Cai, J.: Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. In: *European Conference on Computer Vision*. pp. 370–386. Springer (2024)
14. Chen, Y.C., Ling, S., Chen, Z., Kim, V.G., Gadelha, M., Jacobson, A.: Text-guided controllable mesh refinement for interactive 3d modeling. In: *SIGGRAPH Asia 2024 Conference Papers*. pp. 1–11 (2024)
15. Chen, Y., Mihajlovic, M., Chen, X., Wang, Y., Prokudin, S., Tang, S.: Splatformer: Point transformer for robust 3d gaussian splatting (2025)
16. Chi, Y., Li, X., Huang, Z., Rehg, J.M.: Vinedresser3d: Agentic text-guided 3d editing. *arXiv preprint arXiv:2602.19542* (2026)
17. Chi, Y., Ma, H., Wang, K., Li, J.: Disco3d: Distilling multi-view consistency for 3d scene editing. *arXiv preprint arXiv:2508.01684* (2025)
18. Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In: *European conference on computer vision*. pp. 628–644. Springer (2016)
19. Dao, T.: FlashAttention-2: Faster attention with better parallelism and work partitioning. In: *International Conference on Learning Representations (ICLR)* (2024)

20. Decatur, D., Lang, I., Aberman, K., Hanocka, R.: 3d paintbrush: Local stylization of 3d shapes with cascaded score distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4473–4483 (2024)
21. DeepMind, G.: Nano banana 2: Combining Pro capabilities with lightning-fast speed. Google Blog (February 2026), <https://blog.google>, technical update on Gemini 3.1 Flash Image
22. Deitke, M., Schwenk, D., Salvador, J., Weihs, L., Michel, O., VanderBilt, E., Schmidt, L., Ehsani, K., Kembhavi, A., Farhadi, A.: Objaverse: A universe of annotated 3d objects. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 13142–13153 (2023)
23. Dinh, N.A., Lang, I., Kim, H., Stein, O., Hanocka, R.: Geometry in style: 3d stylization via surface normal deformation. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 28456–28467 (2025)
24. Dong, S., Ding, L., Huang, Z., Wang, Z., Xue, T., Xu, D.: Interactive3d: Create what you want by interactive 3d generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4999–5008 (2024)
25. Engel, J., Schöps, T., Cremers, D.: Lsd-slam: Large-scale direct monocular slam. In: European conference on computer vision. pp. 834–849. Springer (2014)
26. Erkoç, Z., Gümeli, C., Wang, C., Nießner, M., Dai, A., Wonka, P., Lee, H.Y., Zhuang, P.: Predictor3d: Fast and precise 3d shape editing. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 640–649 (2025)
27. Fei, B., Xu, J., Zhang, R., Zhou, Q., Yang, W., He, Y.: 3d gaussian splatting as new era: A survey. IEEE Transactions on Visualization and Computer Graphics (2024)
28. Feng, X., He, Y., Wang, Y., Yang, Y., Li, W., Chen, Y., Kuang, Z., Fan, J., Jun, Y., et al.: Srgs: Super-resolution 3d gaussian splatting. arXiv preprint arXiv:2404.10318 (2024)
29. Forster, C., Zhang, Z., Gassner, M., Werlberger, M., Scaramuzza, D.: Svo: Semidirect visual odometry for monocular and multicamera systems. IEEE Transactions on Robotics **33**(2), 249–265 (2016)
30. Gao, W., Aigerman, N., Groueix, T., Kim, V., Hanocka, R.: Textdeformer: Geometry manipulation using text guidance. In: ACM SIGGRAPH 2023 conference proceedings. pp. 1–11 (2023)
31. Graves, A.: Long short-term memory. Supervised sequence labelling with recurrent neural networks pp. 37–45 (2012)
32. Haque, A., Tancik, M., Efros, A.A., Holynski, A., Kanazawa, A.: Instruct-nerf2nerf: Editing 3d scenes with instructions. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 19740–19750 (2023)
33. Hong, F., Zhang, M., Pan, L., Cai, Z., Yang, L., Liu, Z.: Avatarclip: Zero-shot text-driven generation and animation of 3d avatars. arXiv preprint arXiv:2205.08535 (2022)
34. Hong, Y., Zhang, K., Gu, J., Bi, S., Zhou, Y., Liu, D., Liu, F., Sunkavalli, K., Bui, T., Tan, H.: Lrm: Large reconstruction model for single image to 3d. arXiv preprint arXiv:2311.04400 (2023)
35. Huang, J., Xu, S., Yu, H., Lee, T.Y.: Gsdeformer: Direct, real-time and extensible cage-based deformation for 3d gaussian splatting (2024), <https://arxiv.org/abs/2405.15491>
36. Innovations, K.: Kiri engine: 3d scanner app for iphone, android, and web. <https://www.kiriengine.com/> (2024)

37. Jaegle, A., Gimeno, F., Brock, A., Vinyals, O., Zisserman, A., Carreira, J.: Perceiver: General perception with iterative attention. In: International conference on machine learning. pp. 4651–4664. PMLR (2021)
38. Kamata, H., Sakuma, Y., Hayakawa, A., Ishii, M., Narihira, T.: Instruct 3d-to-3d: Text instruction guided 3d-to-3d conversion. arXiv preprint arXiv:2303.15780 (2023)
39. Kar, A., Häne, C., Malik, J.: Learning a multi-view stereo machine. *Advances in neural information processing systems* **30** (2017)
40. Karami, M., Mirrokni, V.: Lattice: Learning to efficiently compress the memory. arXiv preprint arXiv:2504.05646 (2025)
41. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics* **42**(4) (July 2023), <https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/>
42. Kobayashi, S., Matsumoto, E., Sitzmann, V.: Decomposing nerf for editing via feature field distillation. *Advances in neural information processing systems* **35**, 23311–23330 (2022)
43. Lee, J.L., Li, C., Lee, G.H.: Disr-nerf: Diffusion-guided view-consistent super-resolution nerf. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20561–20570 (2024)
44. Li, P., Ma, S., Chen, J., Liu, Y., Zhang, C., Xue, W., Luo, W., Sheffer, A., Wang, W., Guo, Y.: Cmd: Controllable multiview diffusion for 3d editing and progressive generation. In: Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers. pp. 1–10 (2025)
45. Li, X., Zhang, Q., Kang, D., Cheng, W., Gao, Y., Zhang, J., Liang, Z., Liao, J., Cao, Y.P., Shan, Y.: Advances in 3d generation: A survey. arXiv preprint arXiv:2401.17807 (2024)
46. Liu, F., Wang, H., Chen, W., Sun, H., Duan, Y.: Make-your-3d: Fast and consistent subject-driven 3d content generation. In: European Conference on Computer Vision. pp. 389–406. Springer (2024)
47. Luma AI, I.: Luma ai. <https://lumalabs.ai/> (2024)
48. Michel, E., Adobe: Project new depth: Go from 2d to 3d in one click | adobe max sneaks. https://www.youtube.com/watch?v=1n_p3a-x3yE (2023), accessed: 2025-11-03
49. Michel, O., Bar-On, R., Liu, R., Benaim, S., Hanocka, R.: Text2mesh: Text-driven neural stylization for meshes. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 13492–13502 (2022)
50. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM* **65**(1), 99–106 (2021)
51. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: A versatile and accurate monocular slam system. *IEEE transactions on robotics* **31**(5), 1147–1163 (2015)
52. Nam, S., Sun, X., Kang, G., Lee, Y., Oh, S., Park, E.: Generative densification: Learning to densify gaussians for high-fidelity generalizable 3d reconstruction. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 26683–26693 (2025)
53. Park, J., Kwon, G., Ye, J.C.: Ed-nerf: Efficient text-guided editing of 3d scene with latent space nerf. arXiv preprint arXiv:2310.02712 (2023)
54. PlayCanvas: Supersplat: 3d gaussian splat editor. <https://supersplat.com/> (2024)
55. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion. arXiv preprint arXiv:2209.14988 (2022)

56. Qi, Z., Yang, Y., Zhang, M., Xing, L., Wu, X., Wu, T., Lin, D., Liu, X., Wang, J., Zhao, H.: Tailor3d: Customized 3d assets editing and generation with dual-side images. arXiv preprint arXiv:2407.06191 (2024)
57. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PmLR (2021)
58. Ryu, N., Won, J., Son, J., Gong, M., Lee, J.H., Cho, S.: Elevating 3d models: High-quality texture and geometry refinement from a low-quality model. In: Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers. pp. 1–12 (2025)
59. Schlag, I., Irie, K., Schmidhuber, J.: Linear transformers are secretly fast weight programmers. In: International conference on machine learning. pp. 9355–9366. PMLR (2021)
60. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4104–4113 (2016)
61. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE transactions on Signal Processing **45**(11), 2673–2681 (1997)
62. Sella, E., Fiebelman, G., Hedman, P., Averbuch-Elor, H.: Vox-e: Text-guided voxel editing of 3d objects. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 430–440 (2023)
63. Shazeer, N.: Glu variants improve transformer. arXiv preprint arXiv:2002.05202 (2020)
64. Shen, Y., Ceylan, D., Guerrero, P., Xu, Z., Mitra, N.J., Wang, S., Fröhstück, A.: Supergaussian: Repurposing video models for 3d super resolution. In: European Conference on Computer Vision. pp. 215–233. Springer (2024)
65. Song, H., Choi, S., Do, H., Lee, C., Kim, T.: Blending-nerf: Text-driven localized editing in neural radiance fields. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 14383–14393 (2023)
66. Sun, J., Xie, Y., Chen, L., Zhou, X., Bao, H.: Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 15598–15607 (2021)
67. Sun, Y., Li, X., Dalal, K., Xu, J., Vikram, A., Zhang, G., Dubois, Y., Chen, X., Wang, X., Koyejo, S., et al.: Learning to (learn at test time): Rnns with expressive hidden states. arXiv preprint arXiv:2407.04620 (2024)
68. Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., Hardt, M.: Test-time training with self-supervision for generalization under distribution shifts. In: International conference on machine learning. pp. 9229–9248. PMLR (2020)
69. Teed, Z., Deng, J.: Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. Advances in neural information processing systems **34**, 16558–16569 (2021)
70. Tong, Y., Tian, R., Han, X., Liu, D., Yu, F., Zhang, Y.: Cage-gs: High-fidelity cage based 3d gaussian splatting deformation. arXiv preprint arXiv:2504.12800 (2025)
71. Vachha, C., Haque, A.: Instruct-gs2gs: Editing 3d gaussian splats with instructions (2024), <https://instruct-gs2gs.github.io/>
72. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)

73. Voleti, V., Yao, C.H., Boss, M., Letts, A., Pankratz, D., Tochilkin, D., Laforte, C., Rombach, R., Jampani, V.: Sv3d: Novel multi-view synthesis and 3d generation from a single image using latent video diffusion. In: European Conference on Computer Vision. pp. 439–457. Springer (2024)
74. Wan, Y., Shao, M., Cheng, Y., Zuo, W.: S2gaussian: Sparse-view super-resolution 3d gaussian splatting. arXiv preprint arXiv:2503.04314 (2025)
75. Wang, C., Chai, M., He, M., Chen, D., Liao, J.: Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3835–3844 (2022)
76. Wang, C., Jiang, R., Chai, M., He, M., Chen, D., Liao, J.: Nerf-art: Text-driven neural radiance fields stylization. IEEE Transactions on Visualization and Computer Graphics **30**(8), 4983–4996 (2023)
77. Wang, C., Peng, H.Y., Liu, Y.T., Gu, J., Hu, S.M.: Diffusion models for 3d generation: A survey. Computational Visual Media **11**(1), 1–28 (2025)
78. Wang, D., Zhang, T., Abboud, A., Süssstrunk, S.: Inpaintnerf360: Text-guided 3d inpainting on unbounded neural radiance fields. arXiv preprint arXiv:2305.15094 **2**(4), 9 (2023)
79. Wang, H., Agapito, L.: 3d reconstruction with spatial memory. arXiv preprint arXiv:2408.16061 (2024)
80. Wang, K.A., Shi, J., Fox, E.B.: Test-time regression: a unifying framework for designing sequence models with associative memory. arXiv preprint arXiv:2501.12352 (2025)
81. Wang, Q., Zhang, Y., Holynski, A., Efros, A.A., Kanazawa, A.: Continuous 3d perception model with persistent state. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 10510–10522 (2025)
82. Wen, M., Wu, S., Wang, K., Liang, D.: Intergsedit: Interactive 3d gaussian splatting editing with 3d geometry-consistent attention prior. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 26136–26145 (2025)
83. Wu, J., Bian, J.W., Li, X., Wang, G., Reid, I., Torr, P., Prisacariu, V.: GaussCtrl: Multi-View Consistent Text-Driven 3D Gaussian Splatting Editing. ECCV (2024)
84. Xia, J., Liu, L.: Close-up-gs: Enhancing close-up view synthesis in 3d gaussian splatting with progressive self-training. arXiv preprint arXiv:2503.09396 (2025)
85. Xia, J., Sun, L., Liu, L.: Enhancing close-up novel view synthesis via pseudo-labeling. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 8567–8574 (2025)
86. Xiang, J., Chen, X., Xu, S., Wang, R., Lv, Z., Deng, Y., Zhu, H., Dong, Y., Zhao, H., Yuan, N.J., et al.: Native and compact structured latents for 3d generation. arXiv preprint arXiv:2512.14692 (2025)
87. Xiang, J., Lv, Z., Xu, S., Deng, Y., Wang, R., Zhang, B., Chen, D., Tong, X., Yang, J.: Structured 3d latents for scalable and versatile 3d generation. arXiv preprint arXiv:2412.01506 (2024)
88. Xiang, J., Lv, Z., Xu, S., Deng, Y., Wang, R., Zhang, B., Chen, D., Tong, X., Yang, J.: Structured 3d latents for scalable and versatile 3d generation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 21469–21480 (2025)
89. Xie, S., Wang, Z., Zhu, Y., Pan, C.: Supergs: Super-resolution 3d gaussian splatting via latent feature field and gradient-guided splitting. arXiv preprint arXiv:2410.02571 (2024)
90. Xie, T., Aigerman, N., Belilovsky, E., Popa, T.: Sketch-guided cage-based 3d gaussian splatting deformation. arXiv preprint arXiv:2411.12168 (2024)

91. Xie, T., Zong, Z., Qiu, Y., Li, X., Feng, Y., Yang, Y., Jiang, C.: Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4389–4398 (2024)
92. Xu, T., Chen, J., Chen, P., Zhang, Y., Yu, J., Yang, W.: Tiger: Text-instructed 3d gaussian retrieval and coherent editing. arXiv preprint arXiv:2405.14455 (2024)
93. Xu, Y., Shi, Z., Yifan, W., Chen, H., Yang, C., Peng, S., Shen, Y., Wetzstein, G.: Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. In: European Conference on Computer Vision. pp. 1–20. Springer (2024)
94. Yoon, Y., Yoon, K.J.: Cross-guided optimization of radiance fields with multi-view image super-resolution for high-resolution novel view synthesis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12428–12438 (2023)
95. Yu, X., Zhu, H., He, T., Chen, Z.: Gaussiansr: 3d gaussian super-resolution with 2d diffusion priors. arXiv preprint arXiv:2406.10111 (2024)
96. Zeng, L., Zhao, B., Hu, J., Shen, X., Dang, Z., Bao, H., Cui, Z.: Gaussianupdate: Continual 3d gaussian splatting update for changing environments. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 25800–25809 (2025)
97. Zhang, H., Feng, Y., Kulits, P., Wen, Y., Thies, J., Black, M.J.: Text-guided generation and editing of compositional 3d avatars. arXiv preprint arXiv:2309.07125 (2023)
98. Zhang, K., Bi, S., Tan, H., Xiangli, Y., Zhao, N., Sunkavalli, K., Xu, Z.: Gs-lrm: Large reconstruction model for 3d gaussian splatting. European Conference on Computer Vision (2024)
99. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
100. Zhang, T., Bi, S., Hong, Y., Zhang, K., Luan, F., Yang, S., Sunkavalli, K., Freeman, W.T., Tan, H.: Test-time training done right. arXiv preprint arXiv:2505.23884 (2025)
101. Zhang, T., Kuang, Z., Jin, H., Xu, Z., Bi, S., Tan, H., Zhang, H., Hu, Y., Hasan, M., Freeman, W.T., et al.: Relitlrm: Generative relightable radiance for large reconstruction models. arXiv preprint arXiv:2410.06231 (2024)
102. Zhang, X., Bi, S., Sunkavalli, K., Su, H., Xu, Z.: Nerfusion: Fusing radiance fields for large-scale scene reconstruction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5449–5458 (2022)
103. Zhang, Y., Zhang, L., Ma, R., Cao, N.: Texverse: A universe of 3d objects with high-resolution textures. arXiv preprint arXiv:2508.10868 (2025)
104. Zhou, S., Chang, H., Jiang, S., Fan, Z., Zhu, Z., Xu, D., Chari, P., You, S., Wang, Z., Kadambi, A.: Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21676–21685 (2024)
105. Zhu, Z., Peng, S., Larsson, V., Cui, Z., Oswald, M.R., Geiger, A., Pollefeys, M.: Nicer-slam: Neural implicit scene encoding for rgb slam. In: 2024 International Conference on 3D Vision (3DV). pp. 42–52. IEEE (2024)
106. Zhuang, J., Kang, D., Cao, Y.P., Li, G., Lin, L., Shan, Y.: Tip-editor: An accurate 3d editor following both text-prompts and image-prompts. ACM Transactions on Graphics (TOG) **43**(4), 1–12 (2024)

107. Zhuang, J., Wang, C., Lin, L., Liu, L., Li, G.: Dreameditor: Text-driven 3d scene editing with neural fields. In: SIGGRAPH Asia 2023 Conference Papers. pp. 1–10 (2023)