Painting with 3D Gaussian Splat Brushes

KARRAN PANDEY, University of Toronto, Canada
ANITA HU, NVIDIA, Canada
CLEMENT FUJI TSANG, NVIDIA, Canada
OR PEREL, NVIDIA, Canada
KARAN SINGH, University of Toronto, Canada
MARIA SHUGRINA, NVIDIA, Canada

Fig. 1. Our method explores the creative affordances of painting with real-world texture-geometry content, represented as 3D Gaussian Splats. Our novel technique enables 3D artists to source both their brushes and their canvases from in-the-wild captures, and to swiftly remix highly realistic content through direct and controllable painting for fast 3D prototyping and ideation. Here we use a set of Gaussian splat brushes to embellish this small patch of a local park.

We explore interactive painting on 3D Gaussian splat scenes and other surfaces using 3D Gaussian splat brushes, each containing a chunk of realistic texture-geometry that make capture representations so appealing. The suite of brush capabilities we propose enables 3D artists to capture and then remix real world imagery and geometry with direct interactive control. In particular, we propose a set of algorithms for 1) selecting subsets of Gaussians as a brush pattern interactively, 2) applying the brush interactively to the same or other 3DGS scenes or other 3D surfaces using stamp-based painting, 3) using an inpainting Diffusion Model to adjust stamp seams for seamless and realistic appearance. We also present an ensemble of artistic brush parameters, resulting in a wide range of appearance options for the same brush. Our contribution is a judicious combination of algorithms, design features and creative affordances, that together enable the first prototype implementation of interactive brush-based painting with 3D Gaussian splats. We evaluate our system by showing compelling results on a diverse set of 3D scenes; and a user study with VFX/animation professionals, to validate system features, workflow, and potential for creative impact. Code and data for this paper can be accessed from splatpainting.github.io.

CCS Concepts: • **Computing methodologies** → **Graphics systems and interfaces**; *Volumetric models*; *Rendering*.

Additional Key Words and Phrases: 3D painting, Gaussian splats

ACM Reference Format:
Karran Pandey, Anita Hu, Clement Fuji Tsang, Or Perel, Karan Singh, and Maria Shugrina. 2025. Painting with 3D Gaussian Splat Brushes. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '25), August 10–14, 2025, Vancouver, BC, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3721238.3730724

1 Introduction

The real world abounds with visual richness and diversity, and has always been an important source of creative inspiration for digital 3D authoring. Modern multi-view 3D capture algorithms, such as NERFs [Mildenhall et al. 2021] and 3D Gaussian splats (3DGS) [Kerbl et al. 2023], have progressed in speed, quality and capabilities, but their practical applications remain largely confined to novel view synthesis, in part due to a lack of versatile interaction and editing tools. Emerging work is exploring high-level editing of 3D capture representations, for example through text-based generative AI interfaces [Chen et al. 2024b,a; Haque et al. 2023]. Existing techniques however, lack direct artist control over the editing process, a requirement for most real-life use cases, where the 3D designer has

Authors' Contact Information: Karran Pandey, University of Toronto, Toronto, Canada, karran@cs.toronto.edu; Anita Hu, NVIDIA, Toronto, Canada, anitah@nvidia.com; Clement Fuji Tsang, NVIDIA, Toronto, Canada, cfujitsang@nvidia.com; Or Perel, NVIDIA, Toronto, Canada, orr.perel@gmail.com; Karan Singh, University of Toronto, Toronto, Canada, karan@dgp.toronto.edu; Maria Shugrina, NVIDIA, Toronto, Canada, shumash@gmail.com.

SIGGRAPH Conference Papers '25, Vancouver, BC, Canada

ACM ISBN 979-8-4007-1540-2/25/08
https://doi.org/10.1145/3721238.3730724

specific goals in mind, for example 3D scene prototyping, architectural visualization and game design. Our paper is the first to explore interactive painting on 3D Gaussian splat scenes and other surfaces using 3D Gaussian splat brushes, each containing a chunk of realistic texture-geometry that make capture representations so appealing. The suite of brush capabilities we propose enables 3D artists to capture and then remix real world imagery and geometry with direct interactive control.

The appeal of 3D captures lies in their effortlessly realistic appearance, difficult and time-consuming to achieve with traditional graphics tools. This realism emerges from the unity of local scene geometry and appearance, optimized to resemble the real world. We argue that 3DGS and derivative point-based representations are uniquely suited to novel direct editing interactions, because they encode both appearance and texture explicitly, allowing spatial manipulation without regard for topology, meshes, textures or structured grids. While it is technically fast and simple to cut, move and shift subsets of Gaussians within the scene, appropriate tools are necessary to allow such manipulation in practice. We propose a set of algorithms for 1) selecting subsets of Gaussians as a brush pattern interactively, 2) applying the brush interactively to the same or other 3DGS scenes or other 3D surfaces using stamp-based painting, 3) using inpainting Diffusion Model to adjust stamp seams for seamless and realistic appearance. In addition, we present an ensemble of brush parameters available to the artist, resulting in a wide range of appearance options for the same brush.

Our principle contribution is the design and implementation of the first end-to-end system for interactive modeling based on 3D Gaussian splat painting. Our implementation presents novel solutions to a number of sub-problems including:

- designing a set of interactive tools and brush parameters for artistic brush creation and control using 3DGS content;
- computing oriented 3D Gaussian splat brushes for stamp-based painting on 3D surfaces represented as meshes or 3DGS scenes;
- deforming the splats in a brush stamp to ensure a smooth appearance of the painted 3D stroke;
- producing seamless brush strokes despite overlapping brush stamps, using diffusion inpainting;
- efficient modeling and rendering of brush strokes to facilitate 3DGS painting in real time.

We evaluate our system with numerous painting test cases and conduct a pilot user study with three animation professionals. Following related work (§2), we first explain our 3DGS painting algorithm assuming existence of 3DGS brushes (§3) and then detail brush construction and parameters (§4).

2 Related Work

2.1 Multi-View 3D Capture Representations

NeRF [Mildenhall et al. 2021] and 3D Gaussian splatting (3DGS) [Kerbl et al. 2023] are popular approaches for optimizing volumetric radiance fields from a small collection of images and their corresponding camera information. While both produce highly detailed reconstructions, NeRF variants typically rely on grid support for fast rendering [Fridovich-Keil and Yu et al. 2022; Müller et al. 2022], or global feature decoders for added quality, which makes them more challenging to refine with targeted edits without further optimization. Meanwhile, 3DGS manifestations [Huang et al. 2024; Moenne-Loccoz et al. 2024; Yu et al. 2024b] are Lagrangian in spirit. Their particle-based structure makes them more amenable to local changes, while avoiding the introduction of regional or global artifacts. While both approaches in their vanilla variation suffer from baked lighting, active research in this area [Du et al. 2024; Gao et al. 2025; Moenne-Loccoz et al. 2024] is moving toward relightable 3DGS represenations, amenable to editing. Our work anticipates future advances in this area that will make 3DGS look even more realistic with dynamic lighting effects.

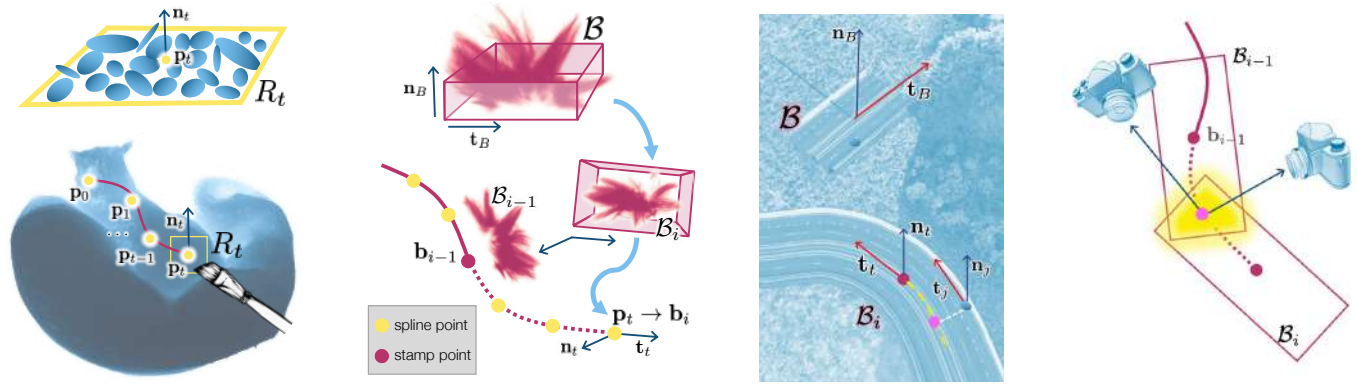
2.2 Editing 3D Captures

The absence of topology makes volumetric radiance fields convenient for global editing applications like stylization conditioned on text [Wang et al. 2022] and images [Liu et al. 2024]. State of the art works have successfully used guidance from powerful generators such as 2D diffusion models [Haque et al. 2023; Koo et al. 2024; Mikaeili et al. 2023]. Nonetheless, grid-based nature of NeRFs make them less flexible than 3DGS, which has seen a large volume of literature directed at primarily AI-based editing, including, but not limited to [Choi et al. 2024; Liu et al. 2024; Palandra et al. 2024; Vachha and Haque 2024; Wang et al. 2024; Wu et al. 2025; Yi et al. 2024]. However, this line of research provides limited means of direct artistic control to Gaussian editing. The only work we are aware of that demonstrates some direct painting-like interaction with 3DGS scenes is GSTex [Rong et al. 2024], which demonstrates rudimentary color change of Gaussians based on color stroke. Direct artist-driven painting is an established creative technique, and our work addresses this gap in 3DGS emerging applications.

2.3 Digital Painting

Digital painting tools are well-established in literature and commercial products, with most tools for texturing designed to periodically stamp an image or stencil over the interactive stroke to achieve diverse appearance. This is true of commercial software for 3D texture painting, such as Mari [Foundry 2023], Substance 3D Painter [Adobe 2023] and 3D Coat [Pilgway 2023]. Similarly to these tools, we adopt a stamp-based approach, but in contrast to prior techniques our brush is not a texture fragment, but contains both geometry and texture. Indeed, scatter-painting features have been employed within established sculpting software like Z-brush, procedural terrain editors such as Gaea [QuadSpinner 2024], World Creator [BiteTheBytes 2024] and game engines Unity [Unity-Technologies 2024] and Unreal Engine [Epic-Games 2025]. While 3D texture brushes have long been used on traditional 3D representations, i.e. meshes, these brushes typically cause a geometry *change* and do not actually stamp geometry, which would be difficult to achieve with e.g. triangular meshes. We aim to show the promise of combined texture-geometry brushes, enabled by the unique nature of 3DGS representation and its unique appearance.

Like other stamp-based approaches working with more complex textural elements, we are faced with the problem of achieving seamless appearance between brush stamps. For example, in the 2D



(a) Spine fitting, normal estimates (§3.2)

(b) Stamp placement (§3.2)

(c) Non-rigid deformation (§3.3)

(d) Inpainting (§3.4)

Fig. 2. **Painting technique:** key aspects of the method in §3.

domain, some user-driven texture retargeting techniques [Lukáč et al. 2015, 2013] can borrow natural texture transitions from an exemplar, and a recent Diffusion Texture Painting adopts a diffusion inpainting model to paint seamless textures on 3D objects with natural transitions [Hu et al. 2024]. Unlike these works, our approach uses 3D texture patches. We further discovered that together with our deformation technique, overlaps in brush stamps of the combined texture-geometry content of 3DGS brush stamps is much less jarring than overlapping image patches of a 2D stroke. Like [Hu et al. 2024], we leverage diffusion inpainting model to improve transitions between stamps, but in our case, we adjust the 3D appearance of 3D patches.

3 Painting with Splats

We will first explain our method of painting on 3DGS scenes and other surfaces, given an existing Gaussian splat brush. In §4, we explore some techniques for constructing the brush itself.

3.1 Preliminaries

The input to our method is an existing 3DGS scene, defined as a set $\mathcal{G}$ of independent Gaussian elements G_i, each defined by its position $\boldsymbol{\mu}_i$, covariance Σ_i, view-dependent color and opacity α_i. We refer to [Kerbl et al. 2023] for the detailed definition of the original Gaussian splats, and our technique readily applies to other Gaussian splat variants, such as [Huang et al. 2024; Moenne-Loccoz et al. 2024], as long as each individual G_i can be transformed by an affine transform T_i, ensuring unchanged high-fidelity appearance under translation, rotation and scaling. In the classic 3DGRT implementation, this entails applying T_i to $\boldsymbol{\mu}_i$, Σ_i as well as the Spherical Harmonics coefficients encoding the directional color effects. In addition, we require the presence of a fast shader [Kerbl et al. 2023] which provides rendered depth, so we may compute $\texttt{first\_hits}(\mathcal{G}, C, \tau_\alpha)$, that for a given camera C outputs the ID of the first hit Gaussian per pixel, subject to a minimum opacity value of τ_α .

The input to our method is any pre-trained 3DGS scene $\mathcal{G}$, and a small 3DGS fragment $\mathcal{B}$ from the same or a different scene, which we will refer to as the brush. The goal of our technique is to enable 3D artists to swiftly remix elements of captured 3D imagery through interactive painting. Just as traditional digital painting software, our brush operates by periodically placing the brush stamp $\mathcal{B}$ along the stroke (§3.2). However, naive stamping results in jarring artifacts, and we develop a stamp deformation technique (§3.3) that ensures more organic appearance of painted 3D content. To push the seamless appearance of strokes further, we propose an automatic stamp inpainting technique using a pre-trained Diffusion Model (§3.4).

3.2 Stamp Placement

Each brush $\mathcal{B}$ comes with an orientation frame $\mathbf{O}_B$ comprising its normal $\mathbf{n}_B$, tangent $\mathbf{t}_B$ and bi-normal $(\mathbf{n}_B \times \mathbf{t}_B)/||\mathbf{n}_B \times \mathbf{t}_B||$ (Fig.2b). Different orientations will produce very different looks, and we give artist full control over these values during brush construction (§4). In order to dynamically place copies of $\mathcal{B}$, which we call stamps $\mathcal{B}_i$, along the stroke during painting, we need to determine each stamp's location in 3D space, and its orientation.

First, we will describe obtaining a point-wise location and normal orientation of a stamp. The preview of this placement is visualized on hover in our system. The tangent orientation is only finalized during painting. Given the current view camera C, we compute $\texttt{first\_hits}(\mathcal{G}, C, \tau_\alpha)$ within a small screen-space rectangle R_t around the cursor location in screen space, resulting in a small set of surface Gaussians $\mathcal{G}_t \in \mathcal{G}$. The average of the means of $\mathcal{G}_t$ becomes the stamp location $\mathbf{p}_t$. The surface patch normal $\mathbf{n}_t$ is estimated via an SVD on the means of Gaussians in $\mathcal{G}_t$, becoming the vertical orientation. Given an artist-specified height offset h, $\mathbf{p}_t$ will be offset along $\mathbf{n}_t$ (here and below, refer to Fig.2). This placement strategy naturally extends to any proxy surface for which we can estimate first hits and normals, such as dense point clouds or 3D meshes - allowing us to support free placement in 3D space via such proxies.

As the artist paints the stroke, we accumulate 3D points $\mathbf{p}_i$ and associated normals $\mathbf{n}_i$, maintaining a dynamic cubic 3D spline $\mathcal{S}$ for the currently painted stroke (Fig.2a). The first control point $\mathbf{p}_0$ is placed based on the rectangle R_0 around the cursor when the stroke is initiated (e.g. through click or stylus touch down). Subsequently, $\mathbf{p}_t$ and $\mathbf{n}_t$ for the current cursor location are computed during every

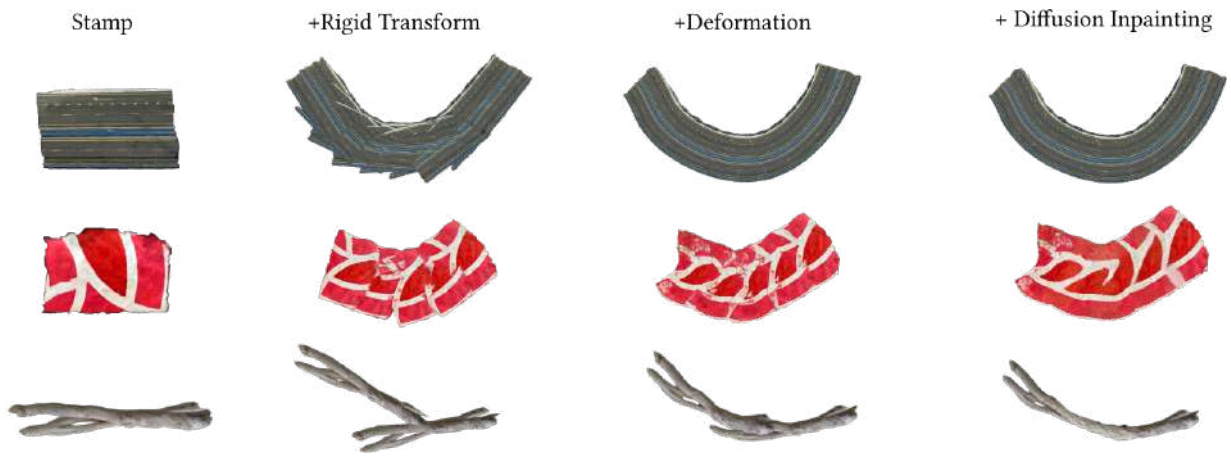


Fig. 3. Ablation of the different steps of our method. (From Left to Right) Stamp Placement (§3.2), Deformation (§3.3) and Inpainting (§3.4).

rendering frame, but is only added to $\mathcal{S}$ if $\|\mathbf{p}_t - \mathbf{p}_{t-1}\| > \tau_S$, for a world-space distance threshold τ_S. In practice, we set τ_S relative to the size of $\mathcal{B}$, and we find that it works robustly across scenes. The spline $\mathcal{S}$ simply determines the geometry of the stroke and its control points are independent of the placed brush stamp locations. We use a cubic spline computation to fit the point set.

In order to evenly space brush stamps $\mathcal{B}_i$ along the stroke, we rely on the arc length parameterization of $\mathcal{S}$ and an artist-specified brush spacing parameter τ_B. At every frame, we compute the arc length distance $s(.,.)$ between the last stamp location $\mathbf{b}_{i-1}$ and spline point under cursor $\mathbf{p}_t$, and place the next stamp if $s(\mathbf{b}_{i-1}, \mathbf{p}_t) > \tau_B$ (Fig.2b). The cubic spline, allows us to also compute a tangent vector $\mathbf{t}_t$. When placing the stamp, we copy the brush Gaussians $\mathcal{B}$ into stamp Gaussians $\mathcal{B}_i$. Then, we compute the best fit rigid rotation to align the brush stamp tangent $\mathbf{t}_B$ and normal $\mathbf{n}_B$ with the $\mathbf{t}_t$ and $\mathbf{n}_t$ under the cursor. The oriented stamp is then translated to place its center at $\mathbf{b}_i \leftarrow \mathbf{p}_t$ offset by h along $\mathbf{n}_t$. No stamp is placed at the initial spline point $\mathbf{p}0$, because the spline tangent cannot yet be computed, but we set saved $\mathbf{b}_0$ to $\mathbf{p}_0$ to bootstrap the stamp placing algorithm. The rigid stamp transform is insufficient for natural appearance, and we next detail a method for non-rigid stamp deformation.

3.3 Stamp Deformation

Although stamp placement specifies the center and orientation for each brush instance, many stamps span a region large enough to encounter curvature along the underlying spline. As a result, we often see undesirable tangential protrusions (see Fig. 3 (Middle Left)). We therefore approximate a non-rigid deformation of the stamp along the spline's curvature using local rigid transformations to each Gaussian splat, following the spline's orientation frame. Our approach is inspired by spatial curve deformation techniques such as "wires" [1998], but adapted for Gaussian splats.

Per-Gaussian Coordinate Transform. We first treat the stamp's center as the origin of its local coordinate system, defined by the stamp's tangent, normal, and binormal ($\mathbf{t}_B$, $\mathbf{n}_B$, $\mathbf{b}_B$). Each Gaussian's center is expressed in this local frame by a 3D coordinate (x, y, z), such that:

$$\mathbf{p}_{\text{stamp}} = x\,\mathbf{t}_B + y\,\mathbf{b}_B + z\,\mathbf{n}_B.$$

When stamping along a spline $\mathcal{S}(a)$ parameterized by arc length a, we compute a spline frame $\big(\mathbf{t}_{\mathcal{S}}(a),\ \mathbf{n}_{\mathcal{S}}(a),\ \mathbf{b}_{\mathcal{S}}(a)\big)$ at each point. To place a Gaussian originally at (x, y, z) in the stamp's frame, we move along $\mathcal{S}$ by x in the tangent direction and then offset by y and z along the local binormal and normal, respectively:

$$\mathbf{p}_{\text{deformed}} = \mathcal{S}(a+x) + y\,\mathbf{b}_{\mathcal{S}}(a+x) + z\,\mathbf{n}_{\mathcal{S}}(a+x).$$

This ensures each Gaussian is rigidly "tracked" along the spline, conforming to its curvature without unwanted scaling.

Orientation Adjustment. In addition to updating Gaussian centers, we also rotate each Gaussian's local axes to match the spline's orientation at its deformed location. Specifically, the stamp's original axes $\big(\mathbf{t}_B, \mathbf{n}_B, \mathbf{b}_B\big)$ are reoriented to align with $\big(\mathbf{t}_{\mathcal{S}}(a+x),\ \mathbf{n}_{\mathcal{S}}(a+x),\ \mathbf{b}_{\mathcal{S}}(a+x)\big)$. This per-Gaussian rotation preserves the shape of each splat and aligns it naturally with the local geometry of the spline.

Because we approximate deformation via local rigid transforms, stamps remain visually coherent even if the spline bends. As a result, large, asymmetric stamps can be painted across curved paths without incurring undue distortions, preserving both the geometric and textural fidelity of the Gaussian splats.

3.4 Diffusion Inpainting

When painting with complex or asymmetric stamps, overlapping regions along a stroke may exhibit visible seams or inconsistent texturing. Optimal positioning and geometric deformation could be insufficient for creating a seamless blend (see red texture pattern in Fig.3). To address this, we introduce a diffusion model-guided inpainting process that modifies the appearance of the overlapping region between stamps to be seamless.

In order to begin the inpanting process, we must first identify the intersecting gaussians and set up camera views which provide the diffusion model with sufficient context for inpainting.

Stamp Overlap and Camera Setup. For each pair of overlapping stamps, we first locate the midpoint on $\mathcal{S}$ between their centers

and compute $\mathcal{S}$'s normal and binormal at that point. Two virtual cameras are then positioned by traveling along these directions—the normal for a top-down view, binormal for a side-on view — and each camera is oriented to look back at the midpoint. The distance is automatically calculated so that both stamps appear fully within the camera's frustum. We render each stamp separately from both camera views to generate visibility masks, and intersect these masks to identify the screen-space overlap between the stamps. We then unproject pixels in the overlap to identify all the gaussians contributing to the overlapping region (yellow region in Fig.2(d)).

Inpainting and Optimization. We then inpaint the 2D screen-space overlap region in each camera view using SD-XL's inpainting checkpoint [Podell et al. 2023], producing a visually plausible transition between stamps. This inpainted result guides an optimization that adjusts the selected Gaussians' opacity and features.

Specfically, let I_{render} and I_{inpaint} be the differentiably rendered and inpainted images, respectively and Let Ω be the set of gaussians in the overlapping region. We optimize the opacity and features of gaussians in Ω based on two loss components: a *visual consistency* loss to inpaint the gaussians and an *opacity* regularization to ensure that we retain a minimum saliency across the optimization.

$$\mathcal{L}_{\text{visual}} = 0.25\,\ell_1\big(I_{\text{render}}, I_{\text{inpaint}}\big) \;+\; 0.75\,\big(1 - \text{SSIM}\big(I_{\text{render}}, I_{\text{inpaint}}\big)\big), \tag{1}$$

where ℓ_1 is the L1 norm (sum of absolute differences) and SSIM measures structural similarity. Let α_i^o be the original opacity of Gaussian $i \in \Omega$, and let α_i be the updated opacity after optimization. We define:

$$\mathcal{L}_{\text{opacity}} = \max\Big(0,\; 0.5 \sum_{i\in\Omega} \alpha_i^o \;-\; \sum_{i\in\Omega} \alpha_i\Big), \tag{2}$$

which penalizes any reduction in total overlap opacity below half of its original value. Finally, the total loss is given by

$$\mathcal{L} \;=\; \mathcal{L}_{\text{visual}} \;+\; \lambda\,\mathcal{L}_{\text{opacity}}, \tag{3}$$

where λ is a hyperparameter controlling the strength of the opacity regularization. We repeat this process for all such stamp overlaps across the stroke. By only optimizing the features and opacity of overlapping gaussians from two relatively independent views, we are able to achieve as seamless as possible transitions without compromising the 3D consistency and integrity of the gaussian stamp and retaining the real-time nature of our application.

Multi-stroke blending. In addition to painting seamless strokes, we can also blend intersections between strokes with diffusion inpainting. This is supported by first allowing the disparate strokes to be placed in the scene without obstruction from each other and blend geometrically. The intersection is then treated as a seam and input to the diffusion inpainting algorithm as before, to create a seamless blend between the strokes (refer to Fig.8 for examples).

4 Brush Creation

We provide a streamlined interface for creating custom Gaussian splat brushes directly from captured scenes. The process consists of two primary steps: (1) selecting a patch of Gaussians to serve as the brush "stamp," and (2) specifying key brush parameters such as orientation, spacing, and height offset.

4.1 Selecting the Brush Patch

Users can select a patch of Gaussians for their brush in two ways: *(i)* via an existing 3D segmenter, or *(ii)* using our custom screen-space bounding box tool, which supports unions, intersections, and removals of selected splats as shown in Fig.4. An optional connected-components mode restricts selections to contiguous regions.

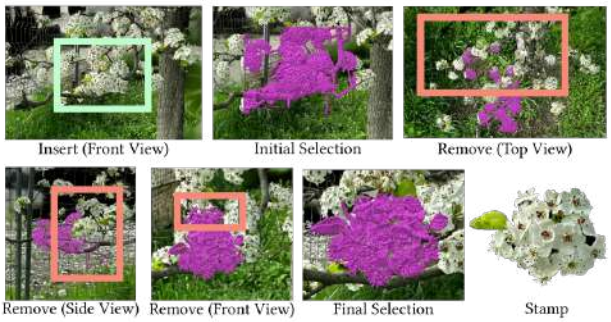


Fig. 4. **Stamp Selection:** Iterative interactive refinement of a selection with screen-space bounding boxes to select a complex flower stamp.

4.2 Brush Parameters

Once the desired Gaussians have been isolated, users can choose to set up the brush using a gizmo that allows them to set the stamp's orientation and scale. The orientation defines the brush's tangent ($\mathbf{t}_B$) and normal ($\mathbf{n}_B$) directions, specifying how the brush will align with surfaces and the underlying spline during painting. Beyond the basic patch selection and orientation, users can also configure the height offset (h): which is the distance above (or below) the surface at which the splats are stamped, and the spacing: which controls how frequently the brush stamp is placed along the spline.

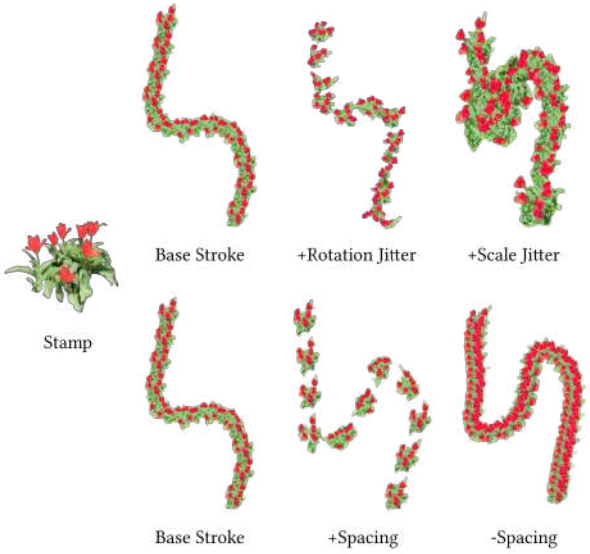


Fig. 5. **Brush Parameters:** Effect of brush parameters on stroke appearance.

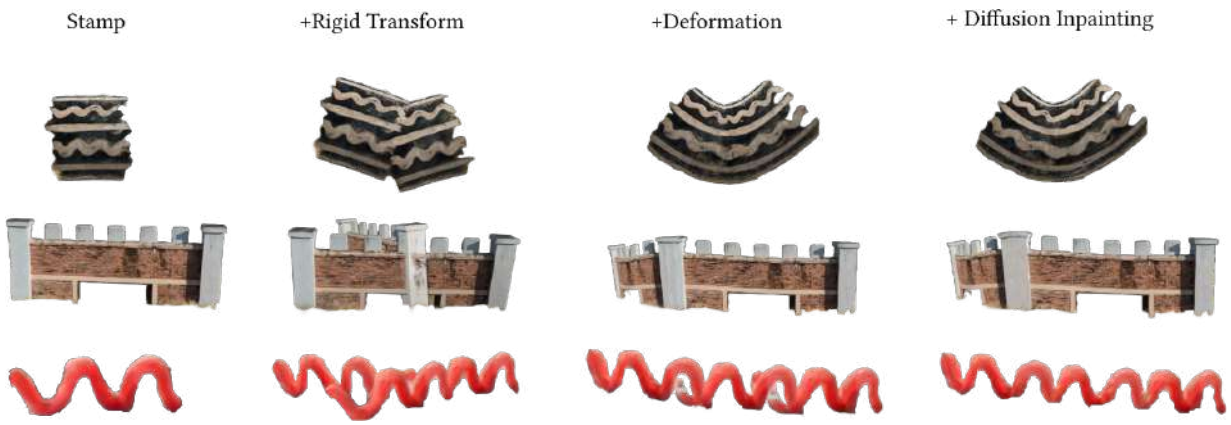


Fig. 6. (From Left to Right): Additional ablations of the different steps of our method. Stamp Placement (§3.2), Deformation (§3.3) and Inpainting (§3.4).

4.3 Random Jittering

To avoid repetitive stamping, random jitter can be introduced for both scale and rotation about the surface normal axis. Users specify a range for these variations, enabling each placement of the brush to differ slightly, producing more organic and varied results.

5 Experiments and Results

5.1 Implementation

All experiments and results presented were generated on a workstation equipped with an NVIDIA RTX 4090 GPU, and all 3DGS scenes used in the paper were captured and processed with Polycam [Polycam 2025]. In our current implementation, the core painting functionality—including spline computations, Gaussian splat brush placement, orientation, non-rigid deformation and basic parameter adjustments—runs in real time for brushes consisting of hundreds of thousands of splats.

The only non-real-time component is the inpainting procedure (§3.4), which is performed parallelly in a background process. As the user continues painting, newly inpainted regions are periodically updated in the scene without interrupting the interactive workflow.

For brush placement, we employ a small screen-space rectangle $\mathcal{R}$ of 15 pixels centered on the cursor to detect initial stamp hits. Additionally, we set the first-hit radius τ_α to be one-tenth of the brush stamp's base side length in all our examples, preventing excessive overlap and facilitating smooth stroke transitions.

5.2 Qualitative Results

In Fig.3 and Fig.6, we show the effect of different aspects of our painting algorithm, including basic stamping (§3.2), non-rigid stamp deformation (§3.3) and diffusion inpainting (§3.4). The surprising finding is that deformation of the splat stamps has the largest effect on the quality of the stroke. This is in contrast to prior art on diffusion texture painting [Hu et al. 2024] (which did not allow painting 3D content), where simple stamping technique was combined with a diffusion model doing the heavy lifting for seamless appearance. In Fig. 8 we illustrate that our blending approach can be extended to inter-stroke interactions as well, providing additional support for iterative painting. In Fig.5, we demonstrate the effect of different brush parameters on the appearance of the stroke. The same captured realistic content can result in wildly different looks, giving artists creative freedom to remix their captured imagery.

Our accompanying gallery Fig. 7 showcases a wide range of use cases, both in terms of diverse brush stamps in style and functionality as well as application scenarios, to demonstrate the flexibility of our approach. In the realm of organic environments, we paint natural-looking foliage (e.g., vines, bushes) with randomized jitter for realistic variation, as well as entire forested landscapes. Structural applications include coherently extending walls, roads, and rail tracks in scanned scenes, while painting in 3D space via surface proxies is exemplified by repairing a broken gingerbread house with a plane mesh and painting birds around a lighthouse using a sphere mesh. We further illustrate texturing capabilities by adding tiger stripes to a cat, detailing fashion garments, and building entire virtual worlds—complete with forests, roads, and houses. These examples collectively underscore the method's versatility across diverse creative and practical use cases.

5.3 User Study

To evaluate the creative potential and usability of our Gaussian splat-based painting tool, we conducted a user study with three professionals (**P1** (Long-time 3D artist, head of production and technical direction with 20 years of experience), **P2** (Technical director specializing in realistic capture with 15 years of experience), **P3** (Technical artist and engineer with 8 years of experience)) from a reputed animation studio. The participants had extensive end-to-end 3D production expertise (modeling, animation, VFX) and experience working with diverse technical pipelines rooted in both traditional modeling and 3D capture, making them ideal evaluators of emerging 3D capture-based creation tools. They provided informed consent; sessions were recorded for analysis but anonymized for reporting.

We began by introducing the study's purpose (a Gaussian splat painting tool) and interviewing participants about their background, experience and their thoughts on the power of capture for creative workflows. After a brief tutorial on the core features of our tool (e.g., brush selection, painting, jitter settings), participants chose between two scanned scenes—one featuring a castle for extending walls and foliage, and another showing a farm with a rail track.

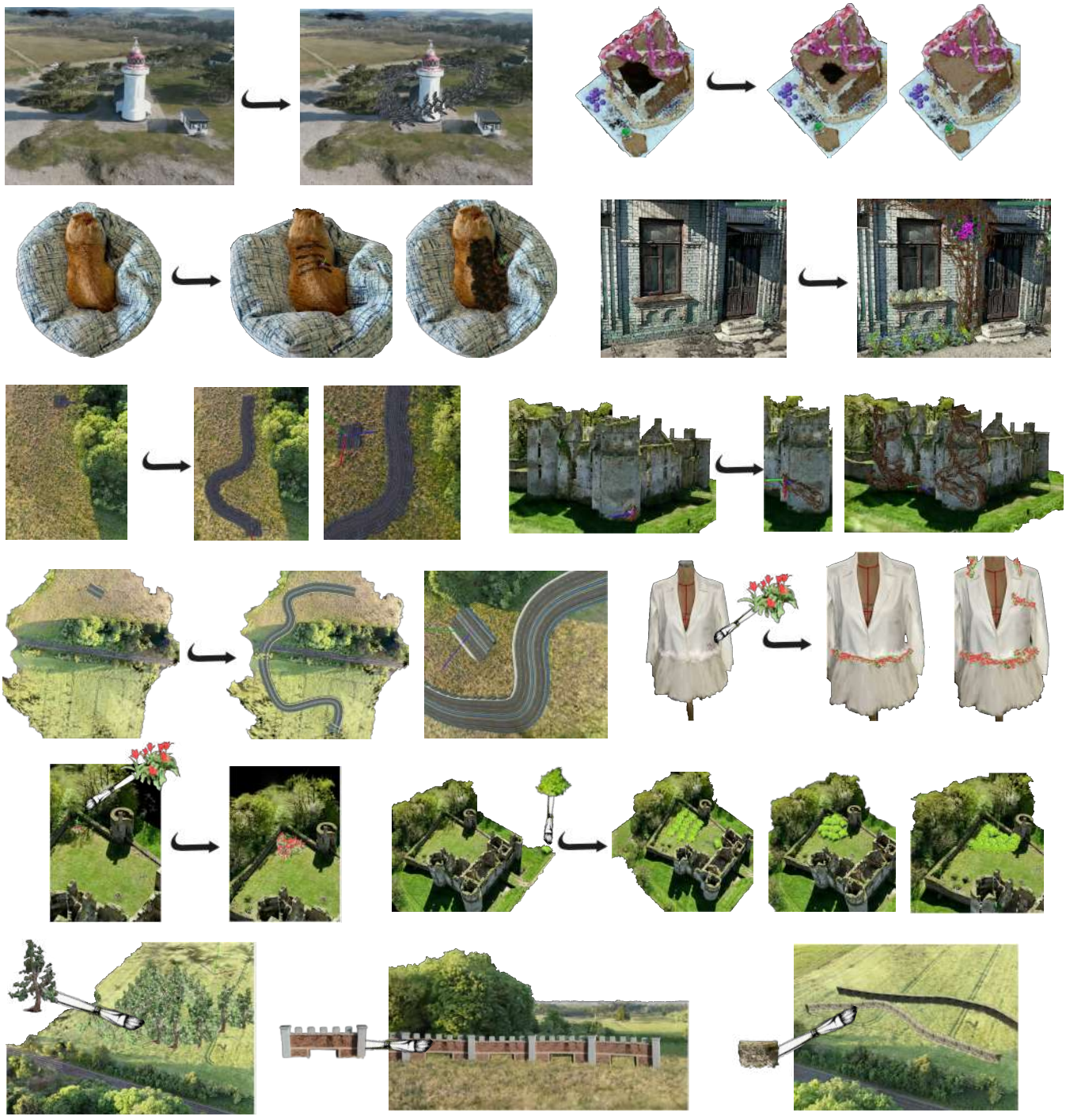


Fig. 7. **Result Gallery**: Our approach has many versatile use cases for quick 3D scene prototyping and design.

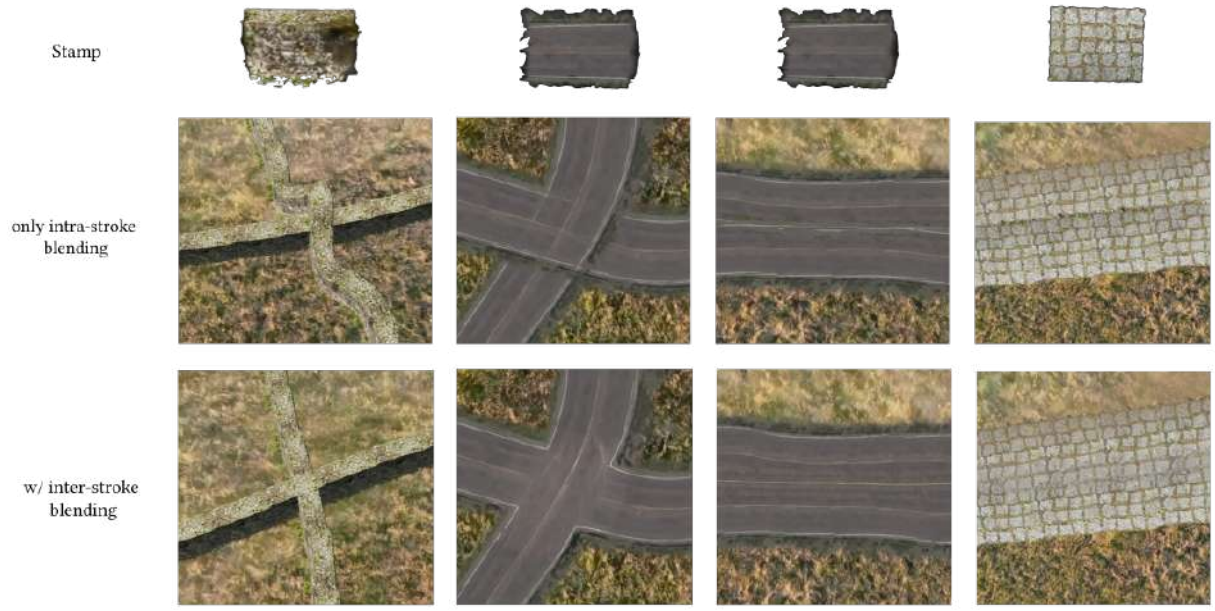


Fig. 8. **Multi-stroke Blending:** Extending our blending approach to multiple strokes. Examples with (bottom) and without (top) multi-stroke blending.

Each scene included a library of ten pre-made brushes. Participants then completed a guided task of either building a small village and forested region near the farm or decorating the castle with walls and foliage, followed by a freeform, open-ended exploration (See Fig. 9 for scenes painted by the participants). We concluded with an interview on overall impressions, potential use cases, and feature requests. We present our findings organized under three key themes: *(1) Features and Creative Potential*, *(2) Extensions*, and *(3) Overall Impressions and Road to Adoption*.

Features and Creative Potential. All participants noted that painting with Gaussian splats could enable rapid, realistic prototyping for concept art and environment design. For instance, **P3** stressed the flexibility of the workflow for *building new worlds*, particularly in the context of text-based AI approaches that may not afford as much direct artist control:

> **P3:** "It's quite easy to create a new world. It's not just a prompt and the AI generates everything; you have more explicit control. You can create worlds really rapidly and easily."

The *immediate feedback of painting* directly onto the scene and the value of brush-based interaction was also noted:

> **P1:** "This paradigm of being able to paint with it is is very important and very useful..the possibility of painting worlds with brushes that you create from real material would be very exciting for a lot of artists. "
>
> **P2:** "You have a step up from existing software like Blender or Unreal... the way you can quickly create roads or coherent structures from a scan is very helpful."

Participants also praised the parameter-based controls (e.g., randomness/jitter), *enabling naturalistic variations*:

> **P1:** "The function, the randomness of the brush, and the fact that you can control that randomness is very, very useful. Anything that has to look a little bit more natural — this is very useful."

Several participants highlighted the *value of easily modifying captured scenes*.

> **P1:** "If we capture an environment and want to modify it quickly, working with photogrammetry is still very painful. But having something that looks realistic and being able to edit it ... that's very valuable."

Others emphasized *urban design* or *architectural visualization* use cases:

> **P2:** "Let's say I scan a park with my drone. I can quickly see what it would look like if we moved the fountain or chairs... It's a good way to design and have a realistic look, faster than redoing everything as a standard mesh."

Another major theme was time-based features for *animation* or *stop-motion*:

> **P1:** "If you can keyframe the painting process, I can imagine some of our artists already doing some very beautiful work with this.. "
>
> **P2:** "I want a timeline at the bottom... frame 1, frame 2, frame 3... as I paint... that would save my life for doing stop-motion with Gaussian splats. The brush could be used to represent movement in time for a stamp too instead of a modeling tool."

Fig. 9. Scenes painted by participants during our pilot study. Apart from the pre-existing castle structure, all other elements (flowers, vines, trees, walls, houses, roads) were added by the participants using pre-made brushes.

Extensions. While participants found the tool promising, they identified several enhancement areas. One prominent request was the ability to *remove or carve existing gaussians*:

> **P1:** "It's nice to be able to add things. If we could also remove or delete..but using the brush..like a carving effect.. that would be really, really, really cool."

Others mentioned semantically *blending* painted gaussians with the underlying scene:

> **P1:** "If you put your blend at 100%, it replaces what's there. At 50%, it does some interesting mix between the the scene and what you're painting - like painting over a tree with flowers adds some of these flowers to its branches and leaves."

P1 suggested extensions to the jitter feature that could make it even more exciting.

> **P1:** "Jitter could be extended to semantic properties of the brush stamp — like the size of a tree trunk or the color of its leaves."

Overall Impressions and Road to Adoption. Our findings suggest that painting directly with real-world Gaussian splats addresses important pain points in rapid scene modification and organic environment creation. Participants envisioned using the tool for:

- *Quick Iterations & Prototyping:* Rapidly augmenting or remixing captured scenes with realistic assets.
- *Naturalistic Variations:* Leveraging randomness to avoid repetition (e.g., foliage, textures).
- *Realistic Structures:* Quickly modeling coherent structures with realistic materials (e.g., roads, train tracks, walls).
- *Stop-Motion/Animation:* Integrating a timeline for keyframed modifications frame by frame.

They found the tool both easy to learn (taking ~10 minutes on average to familiarize themselves with the tool and its features) and use (taking ~5 minutes on average per scene in guided tasks). Our directed questions on the likeliness of adoption received unanimously positive responses, conditional on minor UI, interaction and data handling improvements to better align controls and file formats with popular 3D software conventions familiar to artists. In conclusion, participants were enthusiastic about the potential for Gaussian splat painting to merge the speed of a brush-based workflow with the realism of captured data.

6 Conclusion

We have presented the first method for direct, brush-based painting on 3D Gaussian splat scenes, bridging the gap between the effortless realism of static captured 3D data and the flexible, creative workflows enabled by brush-based interaction. Unifying Gaussian splat-based representations with an intuitive and seamless 3D stamp-based brushing interface, provides artists with the ability to quickly remix geometry and appearance while retaining the rich detail inherent in real-world scans, with a playful interface similar in spirit to the mixing and cloning of 3D meshes [Schmidt and Singh 2010; Takayama et al. 2011].

Our results illustrate the versatility of our approach in handling a variety of use cases: from painting organic foliage and forests to building coherent man-made structures, and texturing animals or garments. These examples demonstrate how Gaussian splat brushes, which tightly integrate texture and geometry, foster efficient iterative design across applications spanning concept art, architectural visualization, and cinematic production. Our pilot study with three animation professionals, further highlighted the tool's potential for rapid prototyping and artistic exploration: emphasizing the immediate value of features such as random jitter for natural variation, direct control over stamp placement, and robust handling of large-scale structural elements via smooth deformations.

Despite these strengths, our framework comes with many limitations. For instance, Gaussian splats lack dynamic relighting once painted. While 3D scenes with baked-in lighting are desirable in several expressive, artistic settings [Yu et al. 2024a], the inability to adapt to changing illumination contexts limits such scenes from wider adoption in 3D cinematic production.

Looking forward, we envision further context-aware painting tools with semantic blending and jitter, and scene segmentation based carving and erasing, for greater artistic freedom. De-lighting and re-lighting brushes, and layering of painted scenes [Yu et al. 2024a] will allow further use of such scenes in a cinematic setting.

We believe this line of research, with the further integration of interactive Gaussian splat painting and emerging diffusion-based geometry-processing techniques, will open new frontiers in expressive 3D painting, allowing artists to directly capture and draw from our visually abundant real-world.

Acknowledgments

We thank the artists who participated in our user study for taking the time to test the prototype extensively and provide valuable feedback. We are grateful to the anonymous reviewers for their insightful comments which inspired significant improvements to the paper. We would like to thank Eloi Champagne, Frank Nadeau and Humbert Hardy from the National Film Board of Canada for insights on artistic applications and use cases and their help in setting up the user studies. Finally, we thank Abhishek Madan, Otman Benchekroun and Esther Lin from the Dynamic Graphics Project for fruitful technical discussions.

References

Adobe. 2023. "Adobe Substance 3D Painter". https://substance3d.adobe.com/documentation/spdoc/paint-brush-34275374.html.

BiteTheBytes. 2024. "World Creator". https://www.world-creator.com/en/index.phtml.

Minghao Chen, Iro Laina, and Andrea Vedaldi. 2024b. DGE: Direct Gaussian 3D Editing by Consistent Multi-view Editing. *arXiv preprint arXiv:2404.18929* (2024).

Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. 2024a. Gaussianeditor: Swift and controllable 3d editing with gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 21476–21485.

Seokhun Choi, Hyeonseop Song, Jaechul Kim, Taehyeong Kim, and Hoseok Do. 2024. Click-gaussian: Interactive segmentation to any 3d gaussians. In *European Conference on Computer Vision*. Springer, 289–305.

Kang Du, Zhihao Liang, and Zeyu Wang. 2024. GS-ID: Illumination Decomposition on Gaussian Splatting via Diffusion Prior and Parametric Light Source Optimization. *arXiv preprint arXiv:2408.08524* (2024).

Epic-Games. 2025. "Unreal Engine". https://www.unrealengine.com/.

Foundry. 2023. Mari - High-resolution digital 3D texture painting. https://www.foundry.com/products/mari.

Fridovich-Keil and Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. 2022. Plenoxels: Radiance Fields without Neural Networks. In *CVPR*.

Jian Gao, Chun Gu, Youtian Lin, Zhihao Li, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. 2025. Relightable 3d gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing. In *European Conference on Computer Vision*. Springer, 73–89.

Ayaan Haque, Matthew Tancik, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. 2023. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 19740–19750.

Anita Hu, Nishkrit Desai, Hassan Abu Alhaija, Seung Wook Kim, and Maria Shugrina. 2024. Diffusion Texture Painting. In *ACM SIGGRAPH 2024 Conference Papers*.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2024. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*. 1–11.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 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/

Juil Koo, Chanho Park, and Minhyuk Sung. 2024. Posterior Distillation Sampling. In *CVPR*.

Kunhao Liu, Fangneng Zhan, Muyu Xu, Christian Theobalt, Ling Shao, and Shijian Lu. 2024. StyleGaussian: Instant 3D Style Transfer with Gaussian Splatting. *arXiv preprint arXiv:2403.07807* (2024).

Michal Lukáč, Jakub Fišer, Paul Asente, Jingwan Lu, Eli Shechtman, and Daniel Sỳkora. 2015. Brushables: Example-based Edge-aware Directional Texture Painting. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 257–267.

Michal Lukáč, Jakub Fišer, Jean-Charles Bazin, Ondřej Jamriška, Alexander Sorkine-Hornung, and Daniel Sỳkora. 2013. Painting by feature: texture boundaries for example-based image creation. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–8.

Aryan Mikaeili, Or Perel, Mehdi Safaee, Daniel Cohen-Or, and Ali Mahdavi-Amiri. 2023. SKED: Sketch-guided Text-based 3D Editing. *ICCV* (2023).

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

Nicolas Moenne-Loccoz, Ashkan Mirzaei, Or Perel, Riccardo de Lutio, Janick Martinez Esturo, Gavriel State, Sanja Fidler, Nicholas Sharp, and Zan Gojcic. 2024. 3D Gaussian Ray Tracing: Fast Tracing of Particle Scenes. *ACM Transactions on Graphics and SIGGRAPH Asia* (2024).

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.* 41, 4, Article 102 (July 2022), 15 pages. doi:10.1145/3528223.3530127

Francesco Palandra, Andrea Sanchietti, Daniele Baieri, and Emanuele Rodolà. 2024. GSEdit: Efficient Text-Guided Editing of 3D Objects via Gaussian Splatting. *arXiv preprint arXiv:2403.05154* (2024).

Pilgway. 2023. 3D Coat. https://3dcoat.com/.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. 2023. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952* (2023).

Polycam. 2025. Polycam - Reality capture for professionals. https://poly.cam/. Accessed: 2025-05-05.

QuadSpinner. 2024. "Gaea". https://quadspinner.com/.

Victor Rong, Jingxiang Chen, Sherwin Bahmani, Kiriakos N Kutulakos, and David B Lindell. 2024. GStex: Per-Primitive Texturing of 2D Gaussian Splatting for Decoupled Appearance and Geometry Modeling. *arXiv preprint arXiv:2409.12954* (2024).

Ryan Schmidt and Karan Singh. 2010. meshmixer: an interface for rapid mesh composition. In *ACM SIGGRAPH 2010 Talks* (Los Angeles, California) *(SIGGRAPH '10)*. Association for Computing Machinery, New York, NY, USA, Article 6, 1 pages. doi:10.1145/1837026.1837034

Karan Singh and Eugene Fiume. 1998. Wires: a geometric deformation technique. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. 405–414.

Kenshi Takayama, Ryan Schmidt, Karan Singh, Takeo Igarashi, Tamy Boubekeur, and Olga Sorkine. 2011. GeoBrush: Interactive Mesh Geometry Cloning. *Computer Graphics Forum* 30, 2 (April 2011), 613–622.

Unity-Technologies. 2024. "Unity". https://unity.com/.

Cyrus Vachha and Ayaan Haque. 2024. Instruct-GS2GS: Editing 3D Gaussian Splats with Instructions. https://instruct-gs2gs.github.io/

Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. 2022. NeRF-Art: Text-Driven Neural Radiance Fields Stylization. *arXiv preprint arXiv:2212.08070* (2022).

Junjie Wang, Jiemin Fang, Xiaopeng Zhang, Lingxi Xie, and Qi Tian. 2024. Gaussianeditor: Editing 3d gaussians delicately with text instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 20902–20911.

Jing Wu, Jia-Wang Bian, Xinghui Li, Guangrun Wang, Ian Reid, Philip Torr, and Victor Adrian Prisacariu. 2025. Gaussctrl: Multi-view consistent text-driven 3d gaussian splatting editing. In *European Conference on Computer Vision*. Springer, 55–71.

Taoran Yi, Jiemin Fang, Junjie Wang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. 2024. GaussianDreamer: Fast Generation from Text to 3D Gaussians by Bridging 2D and 3D Diffusion Models. In *CVPR*.

Emilie Yu, Fanny Chevalier, Karan Singh, and Adrien Bousseau. 2024a. 3D-Layers: Bringing Layer-Based Color Editing to VR Painting. *ACM Trans. Graph.* 43, 4, Article 101 (July 2024), 15 pages. doi:10.1145/3658183

Zehao Yu, Torsten Sattler, and Andreas Geiger. 2024b. Gaussian Opacity Fields: Efficient Adaptive Surface Reconstruction in Unbounded Scenes. *ACM Transactions on Graphics* (2024).