Novel View Synthesis from A Few Glimpses via Test-Time Natural Video Completion

Yan Xu[1] **Yixing Wang**[1] **Stella X. Yu**[1,2]
[1]University of Michigan [2]UC Berkeley
{yxumich,yixingw,stellayu}@umich.edu

Abstract

Given just a few glimpses of a scene, can you imagine the movie playing out as the camera glides through it? That's the lens we take on *sparse-input novel view synthesis*, not only as filling spatial gaps between widely spaced views, but also as *completing a natural video* unfolding through space.

We recast the task as *test-time natural video completion*, using powerful priors from *pretrained video diffusion models* to hallucinate plausible in-between views. Our *zero-shot, generation-guided* framework produces pseudo views at novel camera poses, modulated by an *uncertainty-aware mechanism* for spatial coherence. These synthesized frames densify supervision for *3D Gaussian Splatting* (3D-GS) for scene reconstruction, especially in under-observed regions. An iterative feedback loop lets 3D geometry and 2D view synthesis inform each other, improving both the scene reconstruction and the generated views.

The result is coherent, high-fidelity renderings from sparse inputs *without any scene-specific training or fine-tuning*. On LLFF, DTU, DL3DV, and MipNeRF-360, our method significantly outperforms strong 3D-GS baselines under extreme sparsity. Our project page is at https://decayale.github.io/project/SV2CGS.

1 Introduction

Humans can effortlessly imagine how a scene appears from unseen viewpoints by mentally filling in gaps, by drawing on prior visual experience to infer what's missing. Inspired by this ability, we reinterpret novel view synthesis – a long-standing challenge in computer vision and graphics [8, 21, 11, 46, 32, 70, 48, 18, 27] – as the task of completing a natural video from sparse camera views (Fig. 1). From this perspective, sparse-input novel view synthesis becomes analogous to recovering missing frames in a video captured along an unconstrained camera trajectory. This framing naturally invites the use of powerful generative priors learned from large-scale video data. In particular, pretrained video diffusion models [5, 59], which are trained to synthesize coherent and realistic scene motions, offer a compelling tool for filling in plausible scene content between widely spaced views.

Recently, NeRF [32, 2, 4, 33] and 3D Gaussian Splatting (3D-GS) [18, 65, 13, 15, 28] have significantly advanced novel view synthesis. Unlike NeRF, which represents scenes using an implicit function, 3D-GS models scenes explicitly with a set of 3D Gaussian primitives and renders images through efficient rasterization. 3D-GS achieves photorealistic rendering with substantially faster inference speed, making it a focal point of recent research interest.

However, synthesis from sparse inputs remains difficult. NeRF or 3D-GS methods typically rely on dense input views to accurately constrain the optimization process. In sparse-view settings, occlusions and geometric ambiguities [67] often lead to rendering artifacts and degraded quality. Recent efforts [22, 71, 10, 16, 54, 56] focus more on constrained camera paths (e.g., object-centric or forward-facing views). In contrast, real-world image capture from walking with a handheld smartphone often produces widely spaced, unconstrained views with large occlusions and out-of-view regions (Fig. 1).

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

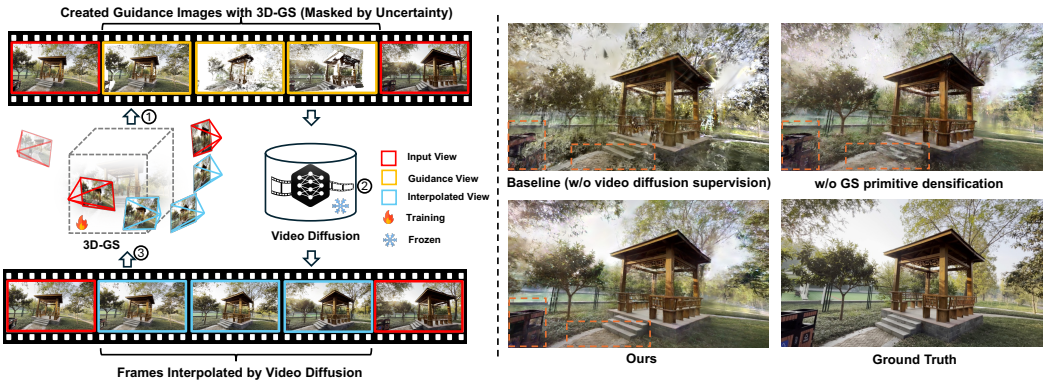


Figure 1: We view sparse-input novel view synthesis as temporal-spatial completion of a natural-looking video. **Left:** Our generation-guided reconstruction pipeline. With the initialized 3D-GS from sparse input views, ① we create guidance images on interpolated poses and estimate their uncertainty, based on the currently optimized 3D-GS. ② Using both guidance images and their uncertainties, we modulate the diffusion score function to interpolate between sparse input views. ③ The interpolated views are used to constrain 3D-GS optimization. **Right:** With our generation-guide reconstruction, the under-observed regions in the inputs are enhanced by the views generated by the diffusion model.

Motivated by the natural video completion perspective and strong priors in pretrained video diffusion models, we propose a **zero-shot, generation-guided reconstruction** pipeline integrating video diffusion with 3D-GS. Our approach defines target camera trajectories between sparse input views and uses video diffusion priors to synthesize plausible intermediate pseudo-views. These views provide supervision to better constrain 3D-GS training, especially in the under-observed regions in the inputs.

To recover missing views along a natural video trajectory, we must generate images at specified camera poses. However, existing video diffusion models [5, 6, 19, 49] are typically conditioned only on the initial frame and produce uncontrolled camera motions. While recent methods [53, 64] introduce trajectory conditioning during training, they still lack guarantees of pose alignment at inference and rely heavily on datasets with camera parameters, limiting generalization and scalability.

We propose a novel *uncertainty-aware modulation mechanism* that couples video diffusion with 3D Gaussian Splatting (3D-GS), enabling accurate, controllable frame interpolation under sparse-view settings. In this setup, 3D-GS provides a consistent 3D representation to guide view synthesis, while synthesized frames serve as pseudo supervision to further refine the 3D-GS model.

Fig. 1 illustrates our overall workflow. Our method begins by initializing 3D-GS from sparse views. After initialization, we interpolate camera poses between sparse inputs and create corresponding guidance images on the interpolated poses by inversely warping pixels from the nearest input view. The warping process is based on the depth maps rendered by the currently optimized 3D-GS. These guidance images are essential to maintaining the content and structural consistency during view interpolation, but may contain missing parts and artifacts due to imperfect 3D-GS depths and occlusion. We thus further model the uncertainty of these guidance images by assessing cross-view consistency in terms of photometry and geometry, and thereby focus the diffusion process more on correcting high-uncertainty regions, while keeping the reliable parts. Using both the guidance images and their associated uncertainties, we adaptively modulate the diffusion process to interpolate between the sparse views. The interpolated pseudo views are then added to the training set of 3D-GS. Furthermore, to improve the scene completeness for 3D-GS, we propose a *Gaussian primitive densification module* to densify the 3D-GS point cloud in under-observed regions using these pseudo views as bridges. The process above is repeated iteratively to refine the 3D-GS reconstruction.

To summarize, our contributions are threefold: **1)** We propose a *zero-shot, generation-guided* 3D-GS pipeline that leverages pretrained video diffusion models to improve novel view synthesis under sparse inputs, particularly in under-observed regions. **2)** We introduce an *uncertainty-aware modulation* mechanism to integrate 3D-GS with video diffusion for controllable pseudo-view generation, and a *Gaussian primitive densification* module to enhance scene completeness. **3)** Our method achieves state-of-the-art performance, with over 2.5 dB PSNR gain on DL3DV and strong results on LLFF and DTU, demonstrating robust generalization. While we primarily use Stable Video Diffusion [5], our framework is agnostic to the diffusion backbone and compatible with alternatives [59, 19].

2 Related Work

Sparse-input Novel View Synthesis. Sparse-input novel view synthesis aims to reconstruct a representation for generating novel views of a scene using a few input images. Although existing training-based methods, *i.e.* NeRF [32] and 3DGS [18], work well with dense inputs, their performance drops significantly with sparse views due to overfitting [41, 50, 34, 12, 43]. Several recent works explore robust novel view synthesis under sparse inputs. One group [7, 34, 16, 47, 44, 20, 71, 60] focuses on imposing additional regularization on views deviating from the training views. For example, GeoAug [7] randomly samples novel views around input frames and constrains rendering to match the input after view warping. Niemeyer *et al.* [34] introduce smooth depth regularization on unseen views. SPARF [47], GeCoNeRF [20], and FewViewGS [60] integrate multi-view correspondence and geometry loss into optimization. However, these methods do not address the fundamental issue of information deficiency in unobserved regions.

Another line of methods explores including priors from pre-trained neural networks [12, 50, 55, 35, 71, 22] for regularization. For example, Jain *et al.* [16] leverage CLIP [38] features to provide semantic guidance. DSNeRF [12] and SparseNeRF [50] use depth regularization from pre-trained depth estimators on known views to guide optimization. More recently, FSGS [71] and DNGaussian [22] extend the similar sprit to 3D-GS training. However, these priors do not directly provide visual supervision for sparse-view NVS like the visual diffusion prior.

Novel View Synthesis with Diffusion Priors. To leverage visual priors for novel view synthesis, several lines of work have emerged. Liu *et al.* [26] use diffusion models to generate pseudo-observations at unseen views, while Wu *et al.* [54] guide the diffusion process using a NeRF representation [62] to synthesize novel views.

To reduce the computational burden of fine-tuning diffusion models, Xiong *et al.* [56] and Wang *et al.* [51] adopt Score Distillation Sampling (SDS) [37] to extract external visual priors. However, these approaches rely on image-based diffusion models and thus fail to fully capture spatiotemporal correlations across views. More recently, Liu *et al.* [25] fine-tuned Stable Video Diffusion [5] to provide view interpolation capability for guiding 3D-GS reconstruction. While this significantly improves performance, it requires substantial computational resources, limiting practical efficiency.

Despite progress in view-conditioned generative models [24, 52, 42, 66], existing methods are either object-centric [24] or struggle to generate photorealistic views [42, 66, 52]. Recent approaches [14, 61, 53, 64] enable coarse camera motion control for video generation from a single frame but lack a consistent 3D representation, which compromises cross-view consistency and reproducibility.

Consequently, how to effectively leverage zero-shot video diffusion priors for novel view synthesis is an important open challenge. The concurrent work [69] is closely related to ours, but it depends on a video diffusion model trained with camera poses [64], and the code was not publicly available at the time of our submission. In contrast, our method can, in principle, be applied to any video diffusion model trained on raw videos, making it more broadly generalizable.

3 Preliminaries – More Details in Appendix

3D Gaussian Splatting (3D-GS) [18] represents 3D scenes explicitly using Gaussian primitives, each defined by mean $\boldsymbol{\mu} \in \mathbb{R}^3$ and covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$: $G(\boldsymbol{x}) = \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)$. Each Gaussian also includes spherical harmonics coefficients $\boldsymbol{c}$ for view-dependent color and an opacity α, enabling expressive appearance modeling. Rendering is performed efficiently via rasterization. After projecting Gaussians to the image plane, pixel colors are computed using alpha compositing: $C_{\text{pix}} = \sum_i \boldsymbol{c}_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j)$, where $\boldsymbol{c}_i$ and α_i denote the color and opacity of the i-th Gaussian, respectively. For depth rendering, $\boldsymbol{c}_i$ is replaced by the z-buffer value.

Stable Video Diffusion (SVD) [5] is an image-to-video diffusion model that generates natural video conditioned on an input image. By default, generation starts from the given image and autonomously evolves, incorporating random camera movements and scene dynamics.

Given a forward diffusion process expressed by $\mathrm{d}\mathbf{x} = f(t)\mathbf{x}\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$, where $\mathbf{x}$ is the noisy latent state at timestamp t, $\mathbf{w}$ denotes the standard Wiener process, and $f(t)$ and $g(t)$ are scalar functions, its reverse process ODE [45] can be expressed as $\mathrm{d}\mathbf{x} = \left[f(t)\mathbf{x} - \frac{1}{2}g^2(\mathbf{x})\nabla_{\mathbf{x}}\log(q_t(\mathbf{x}))\right]\mathrm{d}t$. In the case of the variance exploding (VE) diffusion [45] adopted by Stable Video Diffusion (SVD) [5], it

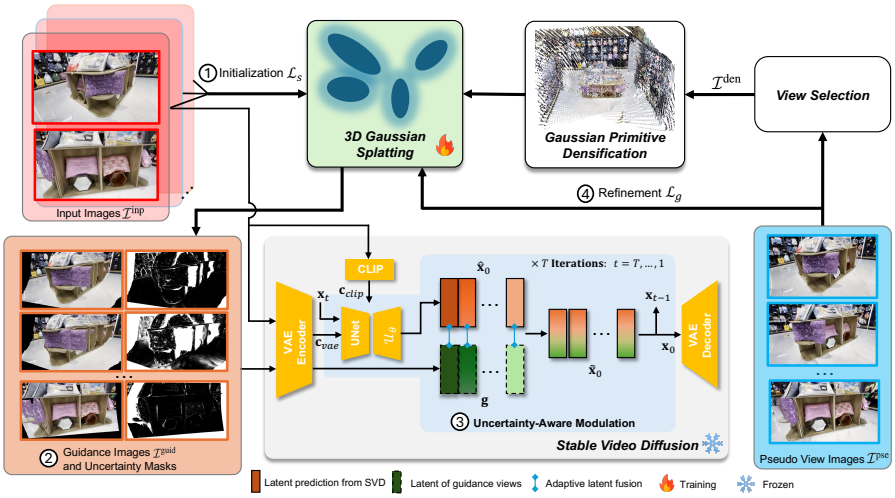


Figure 2: **Overall framework**. After initializing 3D-GS from sparse input images (①), ② we create guidance images (Sec. 4.1.1) and assess their uncertainties (Sec. 4.1.2) based on the current 3D-GS renderings. ③ The guidance images guide the diffusion process through the uncertainty-aware modulation (Sec. 4.1.3). The diffusion process enhances high-uncertain regions while preserving reliable parts. ④ The generated pseudo-view images are then used to densify the Gaussian primitives (Sec. 4.2.1) and to constrain the 3D-GS training (Sec. 4.2.2). For illustration, we show pseudo-view generation from one image pair, though all pairs are processed sequentially in practice.

can be simplified as: $\mathrm{d}\mathbf{x} = \frac{\mathbf{x}-\hat{\mathbf{x}}_0}{\sigma_t}\mathrm{d}\sigma_t$, where the noise of the diffusion process is parameterized as Gaussian noise with a variance of σ_t and $\hat{\mathbf{x}}_0$ is the currently predicted clean video by the network based on the latent state at the previous step. In practice, we can obtain the estimated denoised sample $\mathbf{x}_{t-1}$ at the previous time step by discretizing the diffusion process above:

$$\mathbf{x}_{t-1} = \mathbf{x}_t + \frac{\mathbf{x}_t - \hat{\mathbf{x}}_0}{\sigma_t}(\sigma_{t-1} - \sigma_t). \tag{1}$$

4 Our Test-Time Optimization Approach to Novel View Synthesis

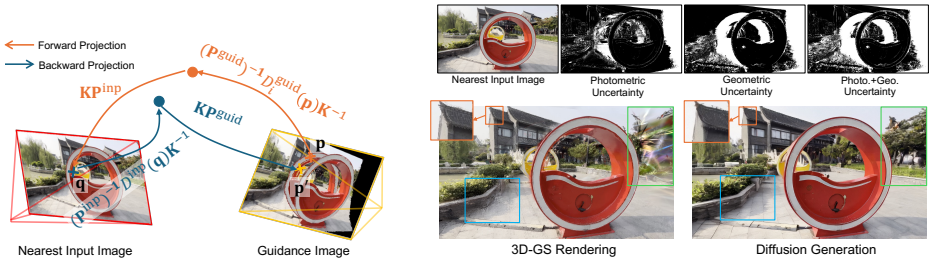
We recast sparse-input novel view synthesis as a test-time natural video completion problem. To this end, we propose an iterative optimization framework that integrates 3D Gaussian Splatting with video diffusion priors to enforce geometric consistency and enhance visual fidelity.

Given a few input views $\mathcal{I}^{\text{inp}}$ and their associated camera poses, we propose a zero-shot, generation-guided reconstruction pipeline that synthesizes novel views by leveraging a pretrained video diffusion model [5] (Fig. 2). The framework consists of four main steps: **1) 3D-GS initialization** from the sparse input views; **2) Guidance feature creation** (Sec. 4.1.1) and their **uncertainty estimation** via a cross-view consistency check(Sec. 4.1.2) based on the current 3D-GS; **3) Uncertainty-aware modulation** of the video diffusion model in generating pseudo views, conditioned on the guidance images and uncertainty masks (Sec. 4.1.3); **4) Refinement of the 3D-GS** by densifying the Gaussian primitives using the generated pseudo-views (Sec. 4.2). Steps **2)–4)** are iteratively performed to progressively improve both the 3D-GS representation and the quality of the diffusion model outputs.

4.1 Pseudo View Generation via Uncertainty-Aware Modulation

Most off-the-shelf video diffusion models lack precise camera control due to the scarcity of datasets with known camera poses. To ensure broad applicability, we design our framework to be compatible with widely available models [5, 59] that are conditioned solely on a single image. Moreover, our approach is theoretically agnostic to variance-exploding diffusion backbones [45].

The modern video diffusion model [5] usually extracts CLIP [38] features $\mathbf{c}_{\text{clip}}$ from the input frame I^{inp} to inform the U-Net of the scene's overall appearance and layout. Simultaneously, the frame is encoded by a VAE encoder to produce contextual features $\mathbf{c}_{\text{vae}}$, which are injected via classifier-free



(a) Illustration of Forward/Backward Projection (b) Example of the Uncertainty Estimates

Figure 3: **Cross-view consistency is evaluated through the forward and backward projections** shown in (a) to estimate the uncertainty of the generated guidance image. As illustrated in (b), regions exhibiting poor cross-view consistency (regions in the boxes) are identified as high-uncertainty areas (brighter), which are subsequently refined by the video diffusion model.

guidance to maintain consistency with the reference frame. At each denoising timestep t, the model denoises a latent video representation $\mathbf{x}_t \in \mathbb{R}^{N\times C\times H\times W}$ using a U-Net $\mathcal{U}_{\boldsymbol{\theta}}(\mathbf{x}_t; \mathbf{c}_{\text{clip}}, \mathbf{c}_{\text{vae}}, t)$, where N, C, H, W are the number of frames, feature and spatial dimensions of the latent, respectively. The U-Net predicts a clean latent $\hat{\mathbf{x}}_0$ from $\mathbf{x}_t$ to update $\mathbf{x}_t$ with Eq. (1), which direct $\mathbf{x}_t$ toward $\hat{\mathbf{x}}_0$. The final denoised latent, $\mathbf{x}_0$, is decoded by the VAE decoder into a video clip.

Our method draws inspiration from diffusion-based image editing techniques [30, 63, 1, 57], particularly SDEdit [30] for its efficiency. Specifically, we propose to modify the original clean latent prediction $\hat{\mathbf{x}}_0$ using the guidance feature $\mathbf{g} \in \mathbb{R}^{N\times C\times H\times W}$ extracted from the guidance images by the VAE encoder. This modification is formulated as an optimization problem applied to each frame i:

$$\widetilde{\mathbf{x}}_0[i] = \arg\min_{\mathbf{x}} \|\mathbf{x} - \hat{\mathbf{x}}_0[i]\|_2^2 + \gamma_{t,i}\|\mathbf{x} - \mathbf{g}[i]\|_2^2, \tag{2}$$

where index $[i]$ denotes the i-th frame channel corresponding to the i-th frame of the generated video, and $\gamma_{t,i} > 0$ is a weighting term that controls the influence of the guidance feature.

The remaining problems are **1)** how to get the proper feature map $\mathbf{g}$ to guide the diffusion model in generating views of desired poses (Sec. 4.1.1) and **2)** how to control $\gamma_{t,i}$ to achieve adaptive modulation (Sec. 4.1.2-4.1.3).

4.1.1 Guidance Feature Creation

The core idea of our approach is to exploit video diffusion priors to infer occluded or missing content from sparse input views. This requires constructing guidance features that are geometrically aligned with the desired target view. To this end, a simple strategy is to render the target views from the current 3D-GS and encode them with the diffusion model's VAE encoder, thereby maintaining 3D consistency. However, this often yields low-fidelity results, as 3D-GS may produce inaccurate color renderings at novel poses during training.

To resolve this, instead of using the 3D-GS to render color images, we create guidance images by inversely warping pixels from their nearest input view, using depth maps rendered by 3D-GS. Concretely, to construct the guidance image I_i^{guid} for the i-th video frame, we first project each pixel $\mathbf{p} \in I^{\text{guid}}$ into the nearest input view $I^{\text{inp}} \in \mathcal{I}^{\text{inp}}$, using the rendered depth map D_i^{guid}, camera intrinsics $\mathbf{K}$, and camera poses $\mathbf{P}^{\text{inp}} \in \mathbb{SE}(3)$ (input view) and $\mathbf{P}_i^{\text{guid}} \in \mathbb{SE}(3)$ (guidance view), to get its corresponding pixel $\mathbf{q}$ in the input image:

$$\mathbf{q} = \mathbf{K}\mathbf{P}^{\text{inp}}(\mathbf{P}_i^{\text{guid}})^{-1} D_i^{\text{guid}}(\mathbf{p})\mathbf{K}^{-1}\mathbf{p}. \tag{3}$$

We fill pixel $\mathbf{p}$ with the color of pixel $\mathbf{q}$ to obtain the guidance image I_i^{guid}. The set of guidance images is denoted as $\mathcal{I}^{\text{guid}} = \{I_i^{\text{guid}}\}_{i=1}^N$, where N is the length of the video clip generated by the video diffusion model in a single pass. The VAE encoder will encode these guidance images to have the corresponding guidance feature maps $\mathbf{g}$ to guide the diffusion process via Eq. (2).

4.1.2 Uncertainty Evaluation from Cross-View Consistency

The constructed guidance images well preserve scene content and structure by adhering to strict multi-view geometric constraints imposed by the 3D-GS representation. However, because 3D-GS is

imperfect during training, especially in under-observed regions, the guidance images may contain missing content or artifacts. To assess the reliability of guidance images, we introduce a strict cyclic consistency check, as illustrated in Fig. 3a. Specifically, in the forward pass, we project each pixel $\mathbf{p}$ in the guidance image to its corresponding pixel $\mathbf{q}$ in the nearest input image using Eq. (3). We then perform a backward projection from $\mathbf{q}$ to the guidance view using the depth map D^{inp} rendered by 3D-GS from the nearest input view: $\mathbf{p}' = \mathbf{K}\mathbf{P}_i^{\text{guid}}(\mathbf{P}^{\text{inp}})^{-1}D^{\text{inp}}(\mathbf{q})\mathbf{K}^{-1}\mathbf{q}$. The uncertainty at pixel $\mathbf{p}$ is then quantified by evaluating both geometric and photometric consistency:

$$U_i(\mathbf{p}) = 1 - \exp\left(-\frac{||\mathbf{p}-\mathbf{p}'||_2^2}{s_1} - \frac{||I_i^{\text{gs}}(\mathbf{p}) - I^{\text{inp}}(\mathbf{q})||_2^2}{s_2}\right), \tag{4}$$

where I_i^{gs} is the 3D-GS rendered image from the view of the i-th guidance image, I^{inp} denotes the nearest input image, and s_1, s_2 are bandwidth parameters controlling the sensitivity to geometric and photometric discrepancies. If the 3D-GS is well constrained at pixel $\mathbf{p}$ and no occlusion is present, the image pixel color $I^{\text{inp}}(\mathbf{q})$ should closely match the color of the 3D-GS rendering $I_i^{\text{gs}}(\mathbf{p})$, and the back-projected position $\mathbf{p}'$ should lie near the original $\mathbf{p}$. This results in low uncertainty. Otherwise, discrepancies in color or geometry increase the uncertainty, as captured by Eq. (4).

4.1.3 Uncertainty-Aware Modulation

Using the uncertainty map, we define $\gamma_{t,i}$ for each pixel in Eq. (2) as:

$$\gamma_{t,i}(\mathbf{p}) = \begin{cases} 0 & U_i(\mathbf{p}) > \delta \text{ or } t < \tau \\ 1/(U_i(\mathbf{p}) + \epsilon) & \text{otherwise} \end{cases}, \tag{5}$$

where δ and τ are threshold hyperparameters and ϵ is a small constant to avoid division by zero. The threshold τ is determined by the overall uncertainty of frame i, measured by $\tau = \frac{k}{HW}\sum_{\mathbf{p}}(U_i(\mathbf{p})) + b$, with k and b as tunable coefficients. This ensures that in uncertain regions, the optimization in Eq. (2) leans towards the diffusion prediction $\hat{\mathbf{x}}_0[i]$, while reliable areas are guided by the features from $\mathbf{g}[i]$. For simplicity, we let $\mathbf{p}$ denote corresponding positions in both image and latent space. In practice, U_i is downsampled via average pooling to match the latent resolution before computing $\gamma_{t,i}$. After computing $\gamma_{t,i}$, we apply Eq. (2) to obtain the fused latent $\tilde{\mathbf{x}}_0[i]$, which is then used in Eq. (1) to update $\mathbf{x}_t$ to $\mathbf{x}_{t-1}$. This reverse sampling step is repeated until the final latent $\mathbf{x}_0$ is obtained, which is then decoded into pseudo-view images via the VAE decoder (see Fig. 2).

4.1.4 Extending to View Interpolation

The above generation pipeline supports view extrapolation from a single input, but may struggle to preserve scene fidelity under large viewpoint shifts. To alleviate this issue, we extend it to view interpolation using two input views as references. We define camera trajectories between them and run the diffusion model forward and backward, conditioned on the start and end images, respectively. At each denoising step, we merge the two latent sequences $\mathbf{x}_{t-1} := \boldsymbol{\beta}\mathbf{x}_{t-1}^{\text{forward}} + (1-\boldsymbol{\beta})R(\mathbf{x}_{t-1}^{\text{backward}})$, where $R(\cdot)$ is the reverse operation along the frame index dimension to align the latent $\mathbf{x}_{t-1}^{\text{backward}}$ to $\mathbf{x}_{t-1}^{\text{forward}}$ in the frame dimension. $\boldsymbol{\beta} \in \mathbb{R}^N$ is the blending weight, with $\boldsymbol{\beta}[i] = (N-i)/(N-1)$ for $i = 1, 2, \ldots, N$, where N is number of interpolated frames between two inputs. See supplementary material for the detailed algorithm.

4.2 3D-GS Optimization Guided by Generation

To constrain the 3D-GS representation, we generate pseudo views by pairing adjacent input images and defining camera trajectories that better cover under-observed regions (see supplementary materials for details). Using the video diffusion model guided by the generated guidance images $\mathcal{I}^{\text{guid}}$, as described in Sec. 4.1, we interpolate between input views to obtain pseudo-view images $\mathcal{I}^{\text{pse}} = \{I_j^{\text{pse}}\}_{j=1}^{pN}$. where p is the number of input image pairs.

4.2.1 Gaussian Primitive Densification

Sparse-input 3D-GS training often yields poor reconstruction in under-observed regions due to limited supervision. To mitigate this, we enhance 3D-GS geometry using generated pseudo-views $\mathcal{I}^{\text{pse}}$ and a dense stereo model [52]. For efficiency, we select a subset of pseudo-views $\mathcal{I}^{\text{den}} \subseteq \mathcal{I}^{\text{pse}}$ whose camera poses yield low inter-frame covisibility, ensuring broad scene coverage with minimal redundancy. These views are used to build a camera graph and optimize a point cloud from stereo predictions. To further improve robustness, we analyze the spatial distribution of the reconstructed points and filter

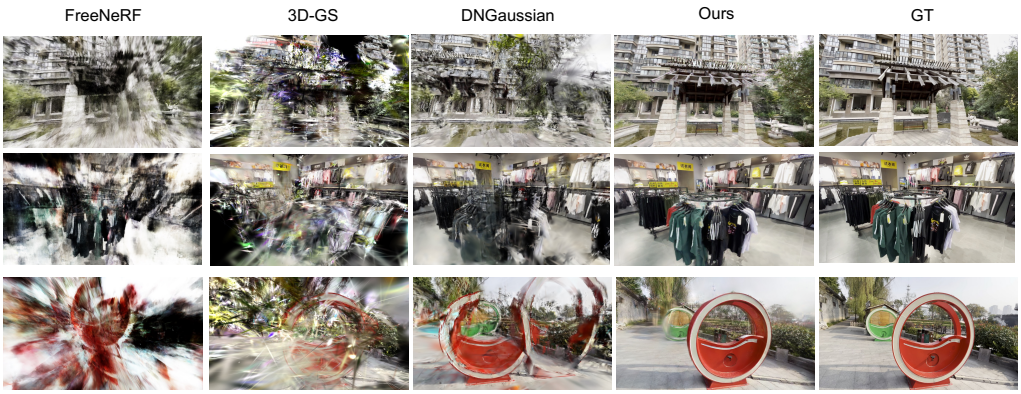


Figure 4: **Qualitative comparison with existing methods on the DL3DV dataset** demonstrates the robustness of our methods against sparse inputs. Leveraging the priors of the video diffusion model, our method renders photorealistic novel views from only 9 input views, while other methods produce noisier, less realistic results.

out those that significantly deviate from the global average distance to neighboring points. Finally, we query existing Gaussian primitives within a fixed radius of each remaining point and only add new Gaussian primitives at positions without nearby primitives to augment the current set. See appendix for more details.

4.2.2 3D Gaussian Splatting Optimization

After densifying the Gaussian primitive set, we optimize the 3D-GS model using both the original inputs and the generated pseudo-views. In each training iteration, one input view and one pseudo-view are sampled for supervision. For the original input views, we apply an L1 loss and a D-SSIM loss , as well as a depth regularization term $\mathcal{L}_{\text{reg}}$ with Pearson correlation similar to [51]: $\mathcal{L}_s = w_1\mathcal{L}_1(I^{\text{gs}}, I^{\text{inp}}) + w_2\mathcal{L}_{\text{D-SSIM}}(I^{\text{gs}}, I^{\text{inp}}) + w_3\mathcal{L}_{\text{reg}}$, where I^{gs} is the rendered image from 3D-GS and I^{inp} denotes the corresponding input image. For the generated pseudo views, we observe that, despite the carefully designed guidance mechanism, some regions still suffer from temporal inconsistency—particularly distant areas with weak geometry or those with fine-grained textures, *e.g.*, grass or tree leaves. To mitigate the negative impact of such inconsistencies on 3D-GS training, we use the LPIPS loss [68] instead of L1 loss. The resulting loss for pseudo-views is:

$$\mathcal{L}_g = w_4\mathcal{L}_{\text{LPIPS}}(I^{\text{gs}}, I^{\text{pse}}) + w_5\mathcal{L}_{\text{D-SSIM}}(I^{\text{gs}}, I^{\text{pse}}) + w_6\mathcal{L}_{\text{reg}}. \tag{6}$$

5 Experiments

5.1 Experiment Settings

Datasets and Metrics. We evaluate our method on LLFF [31], DL3DV [23], DTU [17], and MipNeRF-360 [3] datasets. LLFF consists of 8 forward-facing scenes. Following standard practice [58, 22], we train our model using only 3 input views on this dataset. DL3DV comprises diverse indoor and outdoor scenes, captured by humans walking through scenes. The Mip-NeRF 360 dataset consists of real-world indoor and outdoor scenes designed for evaluating novel view synthesis in large, unbounded environments. Compared to LLFF, DTU, and Mip-NeRF 360, DL3DV offers more diverse scene types and exhibits significantly more complex and dynamic camera motions. We include this dataset to evaluate the robustness of our approach under more realistic and challenging conditions, and our evaluation is conducted on the official test split of DL3DV. To verify the generalizability of our methods and compare with the previous methods, we also test our methods on DTU, an object-centric dataset captured in controlled conditions. For the DTU dataset, we follow the protocol from RegNeRF [22], using 3 training views (IDs 25, 22, and 28) across 15 evaluation scenes. To focus on the object of interest, we mask out the background during evaluation using the provided object masks, consistent with [58, 22]. We apply a downsampling factor of 8 for LLFF and 4 for DTU, aligning with prior work. The rendering quality is assessed using PSNR, SSIM, and LPIPS metrics.

Implementation details. Our pipeline is designed to operate iteratively. In each cycle, we train the 3D-GS model for 10K iterations, followed by an update of the pseudo-view images using the video

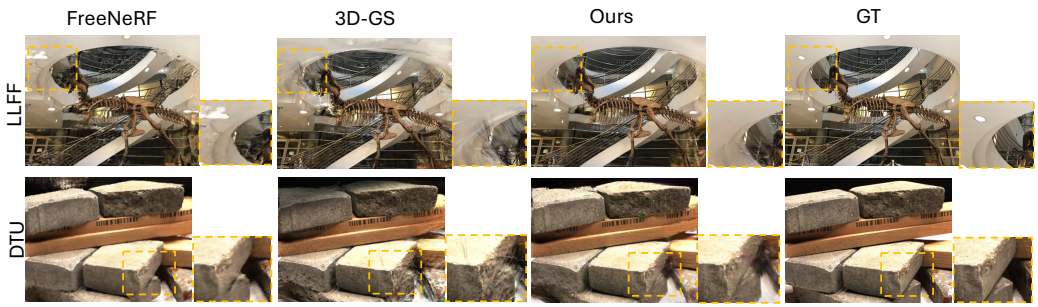


Figure 5: Qualitative comparison with other methods on DTU and LLFF datasets.

Table 1: **Quantitative comparisons with other methods on LLFF, DTU, and MipNeRF-360** demonstrate our state-of-the-art performance and strong generalization ability. **Left**: Quantitative comparison with other methods on the LLFF dataset with 3 training views. **Middle**: Comparison on the DTU dataset with 3 training views. **Right**: Comparison on the MipNeRF-360 dataset with 9 training views. Recent reconstruction-based methods, feed-forward methods, and non-zero-shot methods are included. We color each cell as **best**, **second best**, and **third best**.

LLFF	PSNR↑	SSIM↑	LPIPS↓
Mip-NeRF [2]	16.11	0.401	0.460
3D-GS [18]	17.43	0.522	0.321
DietNeRF [16]	14.94	0.370	0.496
RegNeRF [34]	19.08	0.587	0.336
FreeNeRF [58]	19.63	0.612	0.308
SparseNeRF [50]	19.86	0.624	0.328
FSGS [71]	20.31	0.652	0.288
DNGaussian [22]	19.12	0.591	0.294
IPSM [51]	20.44	0.702	0.207
Ours	20.61	0.705	0.201

DTU	PSNR↑	SSIM↑	LPIPS↓
Mip-NeRF [2]	8.68	0.571	0.353
3D-GS [18]	10.99	0.585	0.313
DietNeRF [16]	11.85	0.633	0.314
RegNeRF [34]	18.89	0.745	0.190
FreeNeRF [58]	19.92	0.787	0.182
SparseNeRF [50]	19.55	0.769	0.201
DNGaussian [22]	18.91	0.790	0.176
SparseGS [56]	18.89	0.834	0.178
Ours	20.51	0.840	0.137

MipNeRF-360	PSNR↑	SSIM↑	LPIPS↓
RegNeRF [34]	13.73	0.193	0.629
FreeNeRF [58]	13.20	0.198	0.635
DNGaussian [22]	12.51	0.228	0.683
MVSplat 360 [9]	14.86	0.321	0.528
ViewCrafter [64]	16.68	0.382	0.551
3DGS-Enhancer [25]	16.22	0.399	0.454
Ours	17.91	0.495	0.435

diffusion model. After each pseudo-view update, we reset the learning rate schedule of 3D-GS before starting the next optimization cycle to avoid overfitting. For the uncertainty estimation in Eq. (4), we set the bandwidth parameters to $s_1 = 100$ and $s_2 = 0.25$. The δ in Eq. (5) is fixed at 0.5 across all experiments. The loss weights are configured as follows: $w_1 = 0.8$, $w_2 = 0.2$, $w_3 = 1.0$, $w_4 = 1.0$, $w_5 = 0.2$, and $w_6 = 1.0$. Additional implementation details are provided in supplementary materials.

5.2 Comparison with Other Methods

We compare our method against state-of-the-art approaches on four benchmark datasets to demonstrate its effectiveness and generalizability across diverse scenarios.

Comparison on LLFF. We evaluate our method on the LLFF dataset captured by a swaying face-forward camera. As shown in Table 1 (left), our method consistently outperforms NeRF-based approaches across all evaluation metrics. When compared to 3D-wGaussian Splatting–based baselines such as FSGS [71] and DNGaussian [22], our method remains competitive, particularly in LPIPS and SSIM scores. This improvement is largely attributed to the additional supervisory signal provided by the pseudo views generated through the video diffusion model. Notably, the LPIPS metric, which correlates more closely with human perceptual similarity than traditional metrics like PSNR, highlights our method's ability to produce visually realistic novel views. Qualitative comparisons are presented in Fig. 5.

Comparison on DTU. To further assess the generalizability of our approach, we evaluate and compare its performance on the DTU dataset. DTU is an object-centric dataset in which each scene contains a centered object against a monotone background. The evaluation results are presented in Table 1 (middle). In this setting, our method still performs well and outperforms other NeRF-based and 3D-GS-based methods. Specifically, our method outperforms the second-best approach by a significant margin in terms of PSNR, SSIM, and LPIPS. While NeRF-based methods also exhibit competitive accuracy in this scenario, they suffer from slow rendering speeds (approximately 0.21 FPS), whereas our 3D-GS-based approach supports real-time rendering at around 430 FPS.

Comparison on DL3DV. We compare with other cutting-edge counterparts on the DL3DV dataset under 3,6, and 9 view settings. Table 2 shows the quantitative comparison results. Apart from the sparse-input 3D-GS methods, we also compare with the non-sparse view methods and NeRF-based methods in Table 2. We outperform previous state-of-the-art methods [22, 71, 51] by a significant

Table 2: **Our method outperforms other test-time optimization methods on the DL3DV dataset**. The results are reported for 3, 6, and 9 training views. We color each cell as **best**, **second best**, and **third best**.

Method	3 Views			6 Views			9 Views		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Mip-NeRF [2]	10.92	0.191	0.618	11.56	0.199	0.608	12.42	0.218	0.600
3DGS [18]	10.97	0.248	0.567	12.34	0.332	0.598	12.99	0.403	0.546
RegNeRF [34]	11.46	0.214	0.600	12.69	0.236	0.579	12.33	0.219	0.598
FreeNeRF [58]	10.91	0.211	0.595	12.13	0.230	0.576	12.85	0.241	0.573
FSGS [71]	12.22	0.296	0.535	13.73	0.429	0.540	15.52	0.468	0.416
DNGaussian [22]	11.10	0.273	0.579	12.67	0.329	0.547	13.44	0.365	0.539
IPSM [51]	11.70	0.279	0.534	12.82	0.332	0.521	13.41	0.361	0.529
Ours	14.62	0.471	0.491	17.35	0.566	0.396	19.19	0.616	0.335

Table 3: **Ablation experiments on the DL3DV test set**. (a) Experiments to show the effectiveness of the proposed components in pseudo-view generation step. (b) Experiments to show the effectiveness of the proposed strategies for 3D-GS optimization.

(a)	PSNR↑	SSIM↑	LPIPS↓
Baseline 3D-GS	16.59	0.502	0.405
w/ GS interpolation	18.59	0.591	0.369
w/ warping interpolation (full)	**19.19**	**0.616**	**0.335**
w/o geometric	18.21	0.583	0.378
w/o photometric	18.93	0.612	0.352

(b)	PSNR↑	SSIM↑	LPIPS↓
w/o point filtering	19.01	0.615	0.343
w/o GS densification	18.23	0.567	0.386
w/o LPIPS loss	18.81	0.597	0.351
Full model	**19.19**	**0.616**	**0.335**

margin in this challenging setting. We observe that although DNGaussian [22] works well in environments with limited scope or with limited camera motions, *e.g.*, object-centric scenarios, it has difficulties in reliably reconstructing the open environment due to the lack of constraints in under-observed regions, as the sparse (a qualitative results shown in Fig. 4). Similarly, FSGS [71] also struggles in this challenging setting, though it achieves slightly better performance compared with DNGaussian because it uses a sparse point cloud for initialization. The recent work IPSM [51] uses an image diffusion model to constrain the 3D-GS by enhancing Score Distillation Sampling (SDS). As shown in Table 2, this method struggles with extremely sparse inputs. This limitation arises because the image diffusion model lacks access to a global scene context, whereas the video diffusion model is able to infer such context from the input reference frame.

Comparison on MipNeRF-360. To evaluate our method on unbounded scenes and ensure a fair comparison with recent feed-forward approaches [9, 64, 25], we further conduct experiments on the Mip-NeRF 360 dataset [3]. Our method consistently outperforms reconstruction-based methods [34, 58, 22] and surpasses state-of-the-art feed-forward approaches [9, 64, 25] by a notable margin. As shown in Fig. 6, although feed-forward methods can hallucinate novel views from sparse inputs through large-scale data training, they often struggle to maintain geometric consistency, fine details, and color fidelity compared to our approach.

5.3 Ablation Study

To validate the effectiveness of our proposed components in the pseudo view generation (Sec. 4.1) and the 3D-GS optimization (Sec. 4.2), we conduct an extensive ablation study on DL3DV.

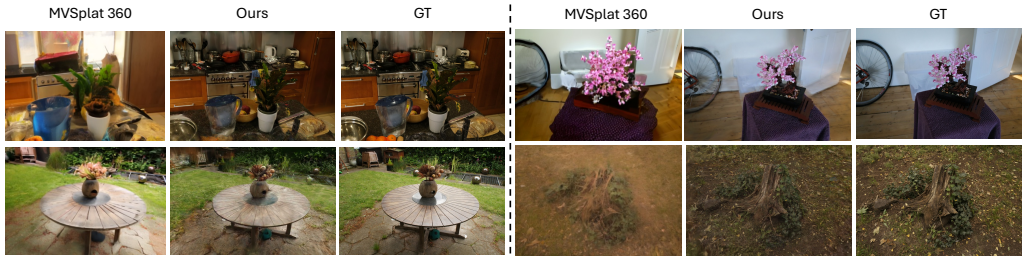


Figure 6: **Our test-time optimization better preserves visual and geometric consistency than the feed-forward approach, MVSplat360**. While feed-forward methods can produce plausible novel views, they often struggle to maintain fidelity to the original scene, whereas our method achieves higher consistency.

Figure 7: **The proposed pseudo-view supervision and primitive densification effectively enhance the novel view synthesis**, especially in under-observed regions from the inputs. Zoom in for a better view.

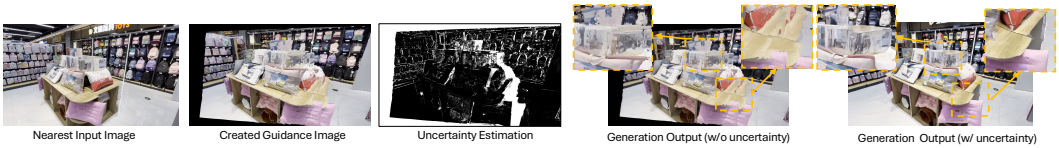


Figure 8: **The estimated uncertainty mask identifies the unreliable parts in guidance images.** The video diffusion model cannot generate faithful images without involving the uncertainty mask.

Effectiveness of uncertainty-aware modulation mechanism. Table 3a compares the baseline 3D-GS trained on sparse views using $\mathcal{L}_s$ with two variants: one using 3D-GS renderings as guidance ("w/ GS interpolation") and one using our warping-based guidance ("w/ warping interpolation"). While GS interpolation improves over the baseline, it underperforms compared to our method due to inaccurate color rendering at novel poses during training.

Fig. 8 shows the effect of uncertainty-aware modulation by comparing diffusion results with and without it, using identical guidance images. We further ablate the geometric and photometric terms in the uncertainty formulation (Eq. (4)), denoted as "w/o geometric" and "w/o photometric." As shown in Table 3a, removing either term noticeably degrades performance.

Effectiveness of Gaussian primitive densification. We ablate the densification step ("w/o GS densification" in Table 3b), observing a significant performance drop, highlighting its role in improving synthesis quality. Fig. 7 shows that densification enhances reconstruction in under-observed regions. Removing the point filtering step ("w/o point filtering") also degrades performance due to depth outliers from the stereo model.

Effectiveness of LPIPS for pseudo view supervision. We replace LPIPS with L1 loss ("w/o LPIPS loss") in Eq. (6), observing a notable performance drop (Table 3b). Despite our guidance strategy, cross-view inconsistencies – especially in distant or textured regions – remain challenging. L1 loss used in vanilla 3D-GS [18] is less robust to such inconsistencies in diffusion-generated pseudo views.

6 Conclusion and Limitation

We introduced a zero-shot, generation-guided pipeline that leverages a pretrained video diffusion model to improve 3D-GS reconstruction from sparse inputs. Intermediate views are synthesized and guided by warped depth-based images and uncertainty-aware modulation. A densification module further enhances scene completeness. Our approach improves photorealism and coverage in sparse settings while maintaining the real-time efficiency of 3D-GS.

Our framework improves sparse-view synthesis but has limitations. It depends on the quality of the pretrained video diffusion model, which may introduce artifacts under extreme views or complex scenes. Iterative training adds overhead compared with vanilla 3D-GS pipelines, and early 3D-GS depth errors can affect guidance quality despite uncertainty modeling, though this impact typically decreases over time.

Societal Impact. This technology can benefit AR/VR, robotics, digital content creation, telepresence, and cultural heritage preservation. However, its computational demands may contribute to a higher carbon footprint.

Acknowledgment. This project was supported, in part, by NSF 2215542, NSF 2313151, and Bosch gift funds to S. Yu at UC Berkeley and the University of Michigan.

References

[1] Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 843–852, 2023.

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields supplemental material.

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5470–5479, 2022.

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19697–19705, 2023.

[5] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

[6] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22563–22575, 2023.

[7] Di Chen, Yu Liu, Lianghua Huang, Bin Wang, and Pan Pan. Geoaug: Data augmentation for few-shot nerf with geometry constraints. In *European Conference on Computer Vision*, pages 322–337. Springer, 2022.

[8] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, page 279–288. Association for Computing Machinery, 1993.

[9] Yuedong Chen, Chuanxia Zheng, Haofei Xu, Bohan Zhuang, Andrea Vedaldi, Tat-Jen Cham, and Jianfei Cai. Mvsplat360: Feed-forward 360 scene synthesis from sparse views. 2024.

[10] Jaeyoung Chung, Jeongtaek Oh, and Kyoung Mu Lee. Depth-regularized optimization for 3d gaussian splatting in few-shot images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 811–820, 2024.

[11] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1996, New Orleans, LA, USA, August 4-9, 1996*, pages 11–20. ACM, 1996.

[12] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12882–12891, 2022.

[13] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2024.

[14] Chen Hou, Guoqiang Wei, Yan Zeng, and Zhibo Chen. Training-free camera control for video generation. *arXiv preprint arXiv:2406.10126*, 2024.

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

[16] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5885–5894, 2021.

[17] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413, 2014.

[18] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

[19] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

[20] Min-Seop Kwak, Jiuhn Song, and Seungryong Kim. Geconerf: Few-shot neural radiance fields via geometric consistency. *arXiv preprint arXiv:2301.10941*, 2023.

[21] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, page 31–42. Association for Computing Machinery, 1996.

[22] Jiahe Li, Jiawei Zhang, Xiao Bai, Jin Zheng, Xin Ning, Jun Zhou, and Lin Gu. Dngaussian: Optimizing sparse-view 3d gaussian radiance fields with global-local depth normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20775–20785, 2024.

[23] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22160–22169, 2024.

[24] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9298–9309, 2023.

[25] Xi Liu, Chaoyi Zhou, and Siyu Huang. 3dgs-enhancer: Enhancing unbounded 3d gaussian splatting with view-consistent 2d diffusion priors. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

[26] Xinhang Liu, Jiaben Chen, Shiu-Hong Kao, Yu-Wing Tai, and Chi-Keung Tang. Deceptive-nerf/3dgs: Diffusion-generated pseudo-observations for high-quality sparse-view reconstruction. In *European Conference on Computer Vision*, pages 337–355. Springer, 2024.

[27] Fan Lu, Yan Xu, Guang Chen, Hongsheng Li, Kwan-Yee Lin, and Changjun Jiang. Urban radiance field representation with deformable neural mesh primitives. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 465–476, 2023.

[28] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024.

[29] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and L Repaint Van Gool. Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471.

[30] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*.

[31] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (ToG)*, 38(4):1–14, 2019.

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

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

[34] Michael Niemeyer, Jonathan T Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5480–5490, 2022.

[35] Avinash Paliwal, Wei Ye, Jinhui Xiong, Dmytro Kotovenko, Rakesh Ranjan, Vikas Chandra, and Nima Khademi Kalantari. Coherentgs: Sparse novel view synthesis with coherent 3d gaussians. In *European Conference on Computer Vision*, pages 19–37. Springer, 2025.

[36] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *The Eleventh International Conference on Learning Representations*.

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

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

[39] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *ArXiv preprint*, 2021.

[40] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[41] Seunghyeon Seo, Yeonjin Chang, and Nojun Kwak. Flipnerf: Flipped reflection rays for few-shot novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22883–22893, 2023.

[42] Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Splatt3r: Zero-shot gaussian splatting from uncalibrated image pairs. 2024.

[43] Nagabhushan Somraj, Adithyan Karanayil, and Rajiv Soundararajan. Simplenerf: Regularizing sparse input neural radiance fields with simpler solutions. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–11, 2023.

[44] Nagabhushan Somraj and Rajiv Soundararajan. ViP-NeRF: Visibility prior for sparse input neural radiance fields. August 2023.

[45] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*.

[46] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *Acm Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[47] Prune Truong, Marie-Julie Rakotosaona, Fabian Manhardt, and Federico Tombari. Sparf: Neural radiance fields from sparse and noisy poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4190–4200, 2023.

[48] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[49] Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[50] Guangcong Wang, Zhaoxi Chen, Chen Change Loy, and Ziwei Liu. Sparsenerf: Distilling depth ranking for few-shot novel view synthesis. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9065–9076, 2023.

[51] Qisen Wang, Yifan Zhao, Jiawei Ma, and Jia Li. How to use diffusion priors under sparse views? *Advances in Neural Information Processing Systems*, 37:30394–30424, 2025.

[52] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20697–20709, 2024.

[53] Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. Motionctrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024.

[54] Rundi Wu, Ben Mildenhall, Philipp Henzler, Keunhong Park, Ruiqi Gao, Daniel Watson, Pratul P Srinivasan, Dor Verbin, Jonathan T Barron, Ben Poole, et al. Reconfusion: 3d reconstruction with diffusion priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21551–21561, 2024.

[55] Jamie Wynn and Daniyar Turmukhambetov. Diffusionerf: Regularizing neural radiance fields with denoising diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4180–4189, 2023.

[56] Haolin Xiong, Sairisheek Muttukuru, Rishi Upadhyay, Pradyumna Chari, and Achuta Kadambi. Sparsegs: Real-time 360 {\deg} sparse view synthesis using gaussian splatting. *arXiv preprint arXiv:2312.00206*, 2023.

[57] Sihan Xu, Yidong Huang, Jiayi Pan, Ziqiao Ma, and Joyce Chai. Inversion-free image editing with language-guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9452–9461, 2024.

[58] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8254–8263, 2023.

[59] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

[60] Ruihong Yin, Vladimir Yugay, Yue Li, Sezer Karaoglu, and Theo Gevers. Fewviewgs: Gaussian splatting with few view matching and multi-stage training. *arXiv preprint arXiv:2411.02229*, 2024.

[61] Meng You, Zhiyu Zhu, Hui Liu, and Junhui Hou. Nvs-solver: Video diffusion model as zero-shot novel view synthesizer. *arXiv preprint arXiv:2405.15364*, 2024.

[62] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4578–4587, 2021.

[63] Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23174–23184, 2023.

[64] Wangbo Yu, Jinbo Xing, Li Yuan, Wenbo Hu, Xiaoyu Li, Zhipeng Huang, Xiangjun Gao, Tien-Tsin Wong, Ying Shan, and Yonghong Tian. Viewcrafter: Taming video diffusion models for high-fidelity novel view synthesis. *arXiv preprint arXiv:2409.02048*, 2024.

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

[66] Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arxiv:2410.03825*, 2024.

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

[68] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[69] Yingji Zhong, Zhihao Li, Dave Zhenyu Chen, Lanqing Hong, and Dan Xu. Taming video diffusion prior with scene-grounding guidance for 3d gaussian splatting from sparse inputs. *arXiv preprint arXiv:2503.05082*, 2025.

[70] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018.

[71] Zehao Zhu, Zhiwen Fan, Yifan Jiang, and Zhangyang Wang. Fsgs: Real-time few-shot view synthesis using gaussian splatting. In *European conference on computer vision*, pages 145–163. Springer, 2025.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction accurately define the scope and assumptions of the problem the paper addresses, and the main claims match the proposed method described in the Method Sec. 4 as well as results presented in Experiment Sec. 5.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discussed the limitations of our work in Sec. 6.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [NA]

Justification: We acknowledge that we do not propose any novel theoretical results that need proofs.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: The details of the proposed method are thoroughly discussed in Sec. 4 and the implementation details are included in Sec. 5.1 and supplementary material. Furthermore, we will release the code upon acceptance.

Guidelines:

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

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

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

   Answer: [Yes]

   Justification: The paper will provide open access to the data and code with instructions to reproduce all experimental results, in the camera ready version upon acceptance.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The training and evaluation details, data selection, and hyperparameters are disclosed in Sec. 5.1 for readers to understand and reproduce the experimental results.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: The experimental results in the paper are not accompanied by any statistical significance tests, as prior work in this area typically only reports aggregate results without such analysis. To ensure comparability and clarity, we follow the reporting conventions in the related literature.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We disclose the the computation resources and computation time we use in supplementary material.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics. The subject of this paper, the proposed method and experiments do not have ethical concerns.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discussed the societal impact at the end of our paper.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: We believe our paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: The data and model backbones we used are cited in the paper. For the code we adopted, we conform with their licenses of use and also properly give credits to them in the corresponding script headers.

    Guidelines:

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

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

13. **New assets**

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

    Answer: [NA]

    Justification: We do not release new assets in the work.

    Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

    Answer: [NA]

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

    Guidelines:

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

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

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

    Answer: [NA]

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: Our paper does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

Appendix

Contents

A Preliminaries on 3D Gaussian Splatting and Diffusion Model

A.1 3D Gaussian Splatting

3D Gaussian Splatting (3D-GS) [18] introduces an explicit representation of 3D scenes using a collection of 3D Gaussian primitives. In this framework, each Gaussian is defined by a mean position $\boldsymbol{\mu} \in \mathbb{R}^3$ and a covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$, which together determine its spatial distribution and shape in 3D space:

$$G(\boldsymbol{x}) = \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right). \tag{S-1}$$

To model the anisotropic shape of each Gaussian, the covariance matrix $\boldsymbol{\Sigma}$ is factorized into a scaling matrix $\boldsymbol{S} = \mathrm{diag}(s_x, s_y, s_z)$ and a rotation matrix $\boldsymbol{R}$, derived from a quaternion:

$$\boldsymbol{\Sigma} = \boldsymbol{R}\boldsymbol{S}\boldsymbol{S}^\top\boldsymbol{R}^\top. \tag{S-2}$$

Beyond its spatial properties, each Gaussian carries additional attributes that define its appearance and transparency. These include spherical harmonics (SH) coefficients $\boldsymbol{c}$ for view-dependent color representation and an opacity parameter α. This combination of spatial and appearance parameters allows for the flexible modeling of complex 3D scenes.

To render a scene, 3D Gaussians are projected onto the 2D image plane through a differentiable splatting process. Given a viewing transformation $\boldsymbol{W}$, each Gaussian is transformed into the camera coordinate system, and its covariance matrix is projected as:

$$\boldsymbol{\Sigma}^{\text{2D}} = \boldsymbol{J}\boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{W}^\top\boldsymbol{J}^\top, \tag{S-3}$$

where $\boldsymbol{J}$ denotes the Jacobian of the projective transformation. The rendering process then computes the contribution of each Gaussian to each pixel, blending their respective colors and opacities through alpha compositing:

$$C_{\text{pix}} = \sum_i \boldsymbol{c}_i\alpha_i\prod_{j=1}^{i-1}(1-\alpha_j). \tag{S-4}$$

Here, $\boldsymbol{c}_i$ and α_i represent the color and opacity of each Gaussian, respectively. The final pixel color is obtained by blending overlapping Gaussian contributions, producing a smooth and high-quality rendering of the 3D scene. To get the depth rendering, we can substitute $\mathbf{c}_i$ with the z-buffer of the corresponding Gaussian.

A.2 Diffusion Models

Stable Video Diffusion (SVD) [5] can be used as an image-to-video diffusion model that generates a natural video conditioned on an input image. By default, the video generation starts with the given image and autonomously evolves, incorporating random camera movements and scene dynamics.

Given a forward diffusion process expressed by $\mathrm{d}\mathbf{x} = f(t)\mathbf{x}\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$, where $\mathbf{x}$ is the noisy latent state at timestamp t, $\mathbf{w}$ denotes the standard Wiener process, and $f(t)$ and $g(t)$ are scalar functions, its reverse process ODE [45] can be expressed as

$$\mathrm{d}\mathbf{x} = \left[f(t)\mathbf{x} - \frac{1}{2}g^2(\mathbf{x})\nabla_{\mathbf{x}}\log(q_t(\mathbf{x}))\right]\mathrm{d}t. \tag{S-5}$$

In the case of the variance exploding (VE) diffusion [45] adopted by Stable Video Diffusion (SVD) [5], Eq. (S-5) can be simplified as:

$$\mathrm{d}\mathbf{x} = \frac{\mathbf{x}-\hat{\mathbf{x}}_0}{\sigma_t}\mathrm{d}\sigma_t, \tag{S-6}$$

where the noise of the diffusion process is parameterized as Gaussian noise with a variance of σ_t and $\hat{\mathbf{x}}_0$ is the currently predicted clean video by the network based on the latent state at the previous step.

In practice, we can obtain the estimated denoised sample $\mathbf{x}_{t-1}$ at the previous time step by discretizing the diffusion process above:

$$\mathbf{x}_{t-1} = \mathbf{x}_t + \frac{\mathbf{x}_t-\hat{\mathbf{x}}_0}{\sigma_t}(\sigma_{t-1}-\sigma_t). \tag{S-7}$$

B Implementation details

B.1 Camera Trajectory Design for View Interpolation

Given sparse-view inputs, we create a set of paired views according to their adjacency and interpolate the camera poses between each pair. For interpolation, we use spherical linear interpolation (slerp) for the rotation component and cubic spline interpolation for the translation component. To enhance coverage of under-observed areas, we repeatedly perturb each interpolated camera pose with random noise to generate a local candidate pool, and then select the candidates with the highest overall uncertainty (as estimated by Eq. (4) in the main paper) from each pool. The selected poses constitute the camera trajectory for view interpolation.

B.2 Algorithm for Pseudo-View Generation

With the paired input views and their interpolated trajectories, we employ the video diffusion model to generate pseudo views between each pair of input images. Since the original video diffusion model [5] is designed to accept only a single input image at a time, we propose a strategy, detailed in Sec. 4.1.4 of the main paper, to effectively incorporate both views for interpolation.

Specifically, we independently encode the start and end views of each image pair using the VAE encoder, and run the denoising U-Net twice: once conditioned on the start view and once on the end view. This process generates two denoised latent sequences, which conceptually correspond to a forward video clip and its reversed version. We denote these two latent sequence at timestep $t-1$ as $\mathbf{x}_{t-1}^{\text{forward}}$ and $\mathbf{x}_{t-1}^{\text{backward}}$, respectively. We then adaptively merge these two latent sequences to provide a better scene understanding for the video diffusion model: $\mathbf{x}_{t-1} = \boldsymbol{\beta}\mathbf{x}_{t-1}^{\text{forward}} + (1-\boldsymbol{\beta})R(\mathbf{x}_{t-1}^{\text{backward}})$, where $R(\cdot)$ is the reverse operation along the frame index dimension to align the latent $\mathbf{x}_{t-1}^{\text{backward}}$ to $\mathbf{x}_{t-1}^{\text{forward}}$ in the frame dimension. $\boldsymbol{\beta} \in \mathbb{R}^N$ is the blending weight, with $\boldsymbol{\beta}[i] = (N-i)/(N-1)$ for $i = 1, 2, \ldots, N$, where N is number of interpolated frames between two inputs. The specific algorithm is provided in Alg. 1.

Algorithm 1 Pseudo-View Interpolation via Uncertainty-Aware Modulation

Require:
$\mathbf{x}_T \in \mathbb{R}^{N\times C\times H\times W}$: Initialized Gaussian noise;
$\mathbf{c}_{\text{clip}}$, $\mathbf{c}'_{\text{clip}}$: CLIP condition for the start and end frame;
$\mathbf{c}_{\text{vae}}$, $\mathbf{c}'_{\text{vae}}$: VAE condition for the start and end frame;
$\mathbf{g} \in \mathbb{R}^{N\times C\times H\times W}$: The guidance image feature encoded by VAE;
σ_t: Gaussian noise variance at timestep t.

for $t = T..1$ **do**
  $\hat{\mathbf{x}}_0 = \mathcal{U}(\mathbf{x}_t; \mathbf{c}_{\text{clip}}, \mathbf{c}_{\text{vae}}, t)$
  $\hat{\mathbf{x}}'_0 = \mathcal{U}(R(\mathbf{x}_t); \mathbf{c}'_{\text{clip}}, \mathbf{c}'_{\text{vae}}, t)$ ▷ $R(\cdot)$ reverses the input in the frame dimension.
  $\widetilde{\mathbf{x}}_0 = \frac{1}{1+\gamma_t}\hat{\mathbf{x}}_0 + \frac{\gamma_t}{1+\gamma_t}\mathbf{g}$. ▷ Vectorized solution of Eq. (2)
  $\widetilde{\mathbf{x}}'_0 = \frac{1}{1+\gamma_t}\hat{\mathbf{x}}'_0 + \frac{\gamma_t}{1+\gamma_t}R(\mathbf{g})$. ▷ Vectorized solution of Eq. (2)
  $\mathbf{x}_{t-1}^{\text{forward}} = \mathbf{x}_t + \frac{\mathbf{x}_t - \widetilde{\mathbf{x}}_0}{\sigma_t}(\sigma_{t-1} - \sigma_t)$.
  $\mathbf{x}_{t-1}^{\text{backward}} = R(\mathbf{x}_t) + \frac{R(\mathbf{x}_t) - \widetilde{\mathbf{x}}'_0}{\sigma_t}(\sigma_{t-1} - \sigma_t)$.
  $\mathbf{x}_{t-1} = \boldsymbol{\beta}\mathbf{x}_{t-1}^{\text{forward}} + (1-\boldsymbol{\beta})R(\mathbf{x}_{t-1}^{\text{backward}})$.
end for
$\mathcal{I}^{\text{pse}}$ = VAE_DEC($\mathbf{x}_0$) ▷ Decode the denoised latent with the VAE decoder to be the output video frames.
return $\mathcal{I}^{\text{pse}}$

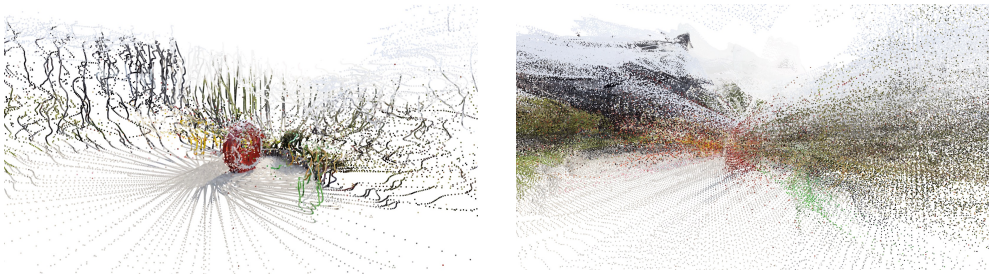


Figure S-1: Effectiveness of the point filtering for Gaussian primitive densification.

B.3 Pseudo-View Selection for Gaussian Primitive Densification

Sparse-input 3D-GS training often yields poor reconstruction in under-observed regions due to limited supervision. To enhance the novel view synthesis in under-observed regions, we propose to densify the Gaussian primitives leveraging the generated pseudo-views $\mathcal{I}^{\text{pse}}$ and a dense stereo model [52].

To improve the efficiency and reduce the redundancy, we select a subset of pseudo views for scene reconstruction. To this end, we define a distance metric that quantifies the covisibility between two views:

$$\text{`CovisibilityScore} = \exp(-\alpha||\mathbf{t}_1 - \mathbf{t}_2||)\exp(-\beta \arccos(\frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{||\mathbf{v}_1||\,||\mathbf{v}_2||})), \tag{S-8}$$

where $\mathbf{t}_1$ and $\mathbf{t}_2$ denote the translation components of the two camera poses, and $\mathbf{v}_1$ and $\mathbf{v}_2$ are their viewing directions, respectively. α and β are parameters to balance the two terms, and are both set to 1 in our case.

We further define a distance metric, *CovisibilityDistance* $= 1 -$ *CovisibilityScore*, to measure the distance between two camera poses. Based on this metric, we perform furthest point sampling over all pseudo-views to select a total of S views. Specifically:

1. We define a set $\mathcal{I}^{\text{den}}$ to contain the views used to densify the Gaussian primitive and initialize it as an empty set;
2. We begin by randomly sampling one view from the set $\mathcal{I}^{\text{pse}}$ and adding it to $\mathcal{I}^{\text{den}}$;
3. We define the distance between a candidate view and the set of views, $\mathcal{I}^{\text{den}}$, as the minimum *CovisibilityDistance* between the candidate and any view already in $\mathcal{I}^{\text{den}}$. We compute this distance for each view in $\mathcal{I}^{\text{pse}} \setminus \mathcal{I}^{\text{den}}$, and select the view with the maximum distance to add to $\mathcal{I}^{\text{den}}$;
4. We repeat step 3 until S views are finally selected, forming the final densification set $\mathcal{I}^{\text{den}}$.

We empirically set $S = 36$, which we found provides sufficient coverage while maintaining computational efficiency.

B.4 Point Cloud Optimization from Multi-view Stereo Estimation

We construct a complete graph over the selected views and jointly optimize their depth maps, guided by the stereo model outputs [52]. Concretely, we treat the depth maps and per-frame scale coefficients as optimizable parameters. Both the optimizable depth maps and the point clouds predicted by the stereo model are projected into the 3D-GS coordinate system using the corresponding camera parameters. For each pair of connected nodes in the graph, we minimize the distance between the projected points from the depth maps and the stereo predictions. The complete graph structure ensures global scale consistency across views. During this process, camera parameters—initialized in the 3D-GS coordinate system—are held fixed. After optimization, the depth maps from all views are fused into a global point cloud.

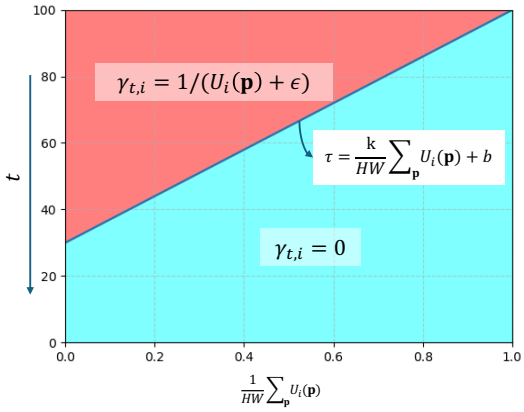


Figure S-2: Illustration of the relation between threshold τ and the uncertainty U_i. The boundary between the red and blue regions is defined by $\tau = \frac{k}{HW}\sum_{\mathbf{p}}(U_i(\mathbf{p})) + b$.

To alleviate the outliers from depth estimation and reduce the computational burden, we uniformly downsample the resultant point clouds and analyze the spatial distribution of the points obtained from the depth maps to filter out those that significantly deviate from the global average distance to neighboring points. We visualize the point clouds with and without this processing step in Fig. S-1.

Finally, we query existing Gaussian primitives within a fixed radius of each remaining point and only add new Gaussian primitives at positions without nearby primitives to augment the current set. The query radius is empirically set to the 85th percentile of the inter-point distances among the current Gaussian primitives.

B.5 More Implementation Details

LLFF. Following FSGS [71], we select every eighth image as the test set, and evenly sample sparse views from the remaining images for training. We utilize 3 views to train all the methods. We follow previous methods to initialize the 3DGS with the point clouds from SfM [40].

DL3DV. We use DL3DV's test set for evaluation and hold every eighth image as our test split, and evenly sample sparse views (*i.e.*, 3, 6, 9 views) from the remaining images for training. We implement the 3D-GS baseline using the point clouds estimated with [52] from the sparse input views.

DTU. For the DTU dataset, we follow the protocol from RegNeRF [22], using 3 training views (IDs 25, 22, and 28) across 15 evaluation scenes. To focus on the object of interest, we mask out the background during evaluation using the provided object masks, consistent with [58, 22].

Our training framework is conducted cyclically. In each cycle, we train the 3D-GS model with 10K iterations and run the video diffusion model to update the pseudo-view images guided by the current 3D-GS. After each update of the pseudo-view images, we reset the learning rate schedule for 3D-GS and start the training of the next cycle. We empirically conduct 3 cycles for our experiments, although more optimization cycles can lead to better performance. In our implementation, we use DPT [39] to predict pseudo depth ground truth for regularization in $\mathcal{L}_{reg}$. For depth regularization on the pseudo views, in the first 30% training iterations, we produce pseudo depth based on the generated pseudo views. For the latter iterations, we generate pseudo-depth based on the color rendering from 3D-GS for better fidelity. For the experiments on LLFF and DTU datasets, the point clouds used for initialization are estimated by SfM [40] using the sparse input views. All experimental results are obtained on a single NVIDIA A40 GPU.

We illustrate the relation between the threshold τ (used in Eq. (5) in the main draft) and the overall uncertainty of an image. During the reverse sampling, the timestamp t decreases from the maximum timestamp, *e.g.* 100, to 0. In the early stage, $t \geq \tau$, the $\gamma_{t,i}$ in Eq. (6) (main paper) is set to $1/(U_i(\mathbf{p}) + \epsilon)$, while after $t < \tau$, $\gamma_{t,i}$ is set to 0. In the red region, the sampling process will rely more on the synthesized guidance images, while in the blue region, the prediction from the U-Net $\mathcal{U}_\theta$

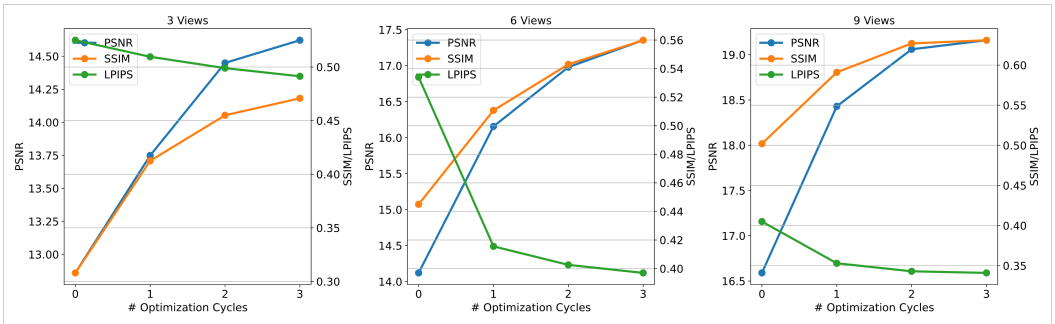


Figure S-3: Novel view synthesis performance across different optimization cycles, evaluated on DL3DV. We report PSNR, SSIM, and LPIPS metrics under varying numbers of input training views. SSIM and LPIPS share the right y-axis due to their similar value ranges.

is adopted. Generally, the larger the overall uncertainty is, the fewer reverse sampling steps will rely on the guidance image I_i^g. By changing k and b, the boundary can be adjusted. We found that our method is relatively robust to the selection of k and b. In our experiment, we set $k = 70$ and $b = 30$.

C Additional Experiments

C.1 Performance Variation as the Optimization Cycle Increases

We evaluated the novel view synthesis performance after different optimization cycles of 3D-GS. Fig. S-3 plots the performance variation on DL3DV. We observed that the performance steadily improves as more cycles of optimization are conducted, which demonstrates that the 3D-GS and the output from the diffusion model can enhance each other based on our proposed uncertainty-aware modulation mechanism. Though more optimization cycles may further improve overall performance, we set the number of optimization cycles to 3 in our experiments for training efficiency.

C.2 Comparison with Methods Based on Image Inpainting

To demonstrate the advantages of using a video diffusion model for view completion, we integrate an image inpainting model [29] into our pipeline as a replacement. Specifically, we use the uncertainty mask to indicate regions for inpainting. Evaluation results of these variants are shown in Table S-1. We implement two versions: (1) directly using the inpainting model's outputs to supervise 3D-GS training ("Baseline 3D-GS+RePaint"), and (2) guiding the inpainting process with 3D-GS renderings using SDS [36]. While image inpainting improves upon the baseline, it remains inferior to our video diffusion-based method, as shown in Table S-1. This is because, in our framework, the video diffusion model leverages intact, paired input views to better infer intermediate content, whereas the inpainting model can only operate on a single, incomplete image for scene hallucination.

	PSNR↑	SSIM↑	LPIPS↓
Baseline 3D-GS+ RePaint [29]	17.45	0.516	0.403
Baseline 3D-GS+SDS[36]	18.12	0.551	0.738
Ours	**19.19**	**0.616**	**0.335**

Table S-1: We replace our video completion module with image-based inpainting techniques and compare the performance of these variants on DL3DV.

C.3 Runtime of Our Pipeline

The 3D-GS training in each optimization cycle takes approximately 30 minutes, while the video diffusion model requires 20 minutes to generate 25 frames on a single NVIDIA A40 GPU. The slow inference speed of the diffusion model is the primary bottleneck. Training efficiency could be improved by distributing 3D-GS and diffusion inference across separate GPUs and running them in

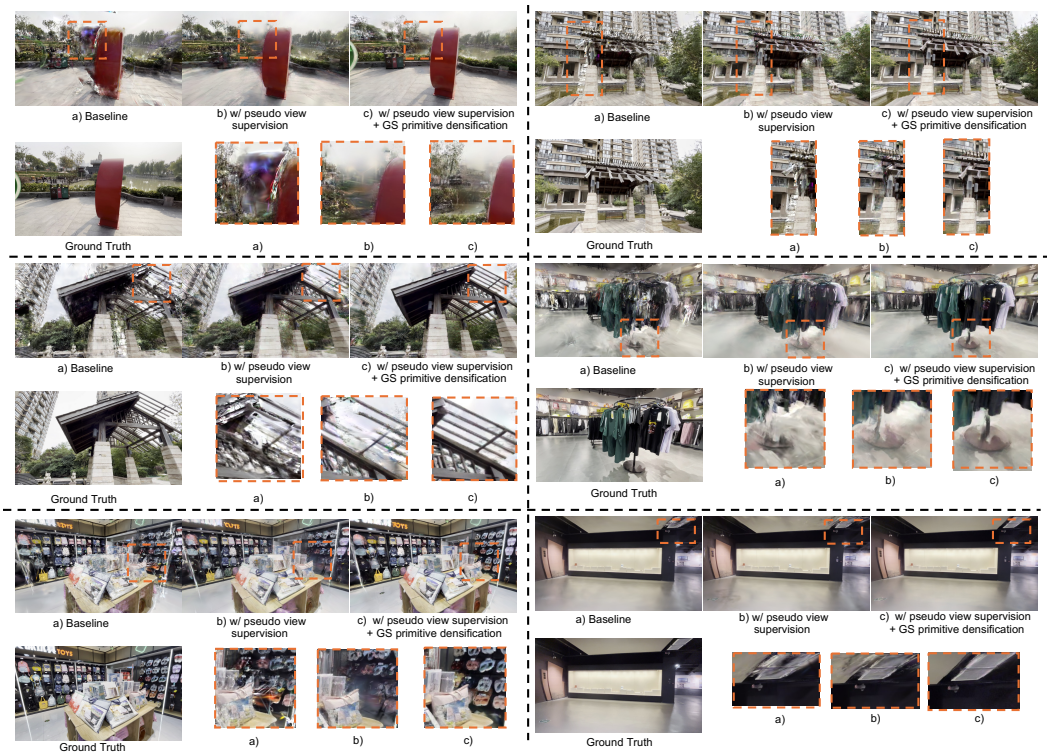


Figure S-4: Qualitative comparisons demonstrate the effectiveness of our generation-guided reconstruction pipeline. The proposed pseudo-view supervision and Gaussian primitive densification significantly enhance training in under-observed regions, leading to more photorealistic novel view synthesis results.

parallel—an optimization we leave for future work. Importantly, the final 3D-GS model maintains real-time rendering performance, achieving over 400 FPS on the LLFF dataset.

C.4 Additional Qualitative Results

Fig. S-4 provides more novel view synthesis results to demonstrate the effectiveness of our proposed modules. We adopt a strong baseline model by initializing the 3D-GS with point clouds constructed by [52]. However, our proposed modules—the pseudo-view supervision and Gaussian primitive densification—can further enhance the novel view synthesis quality, especially in the regions under-observed from the sparse input views.

Fig. S-5 and Fig. S-6 show additional qualitative comparisons on the LLFF and DL3DV datasets. By incorporating constraints from interpolated views generated by the video diffusion model, our method more effectively supervises the 3D-GS representation, especially in under-observed regions. These pseudo views offer direct visual guidance where sparse inputs fall short. In contrast, baseline methods rely solely on sparse inputs, resulting in inferior performance on these challenging datasets.

Fig. S-7 presents additional qualitative comparisons on the DTU dataset. Our method remains robust in this object-centric setting, producing more detailed textures compared to FreeNeRF and vanilla 3D-GS. Notably, these improvements are achieved while maintaining real-time rendering speed.

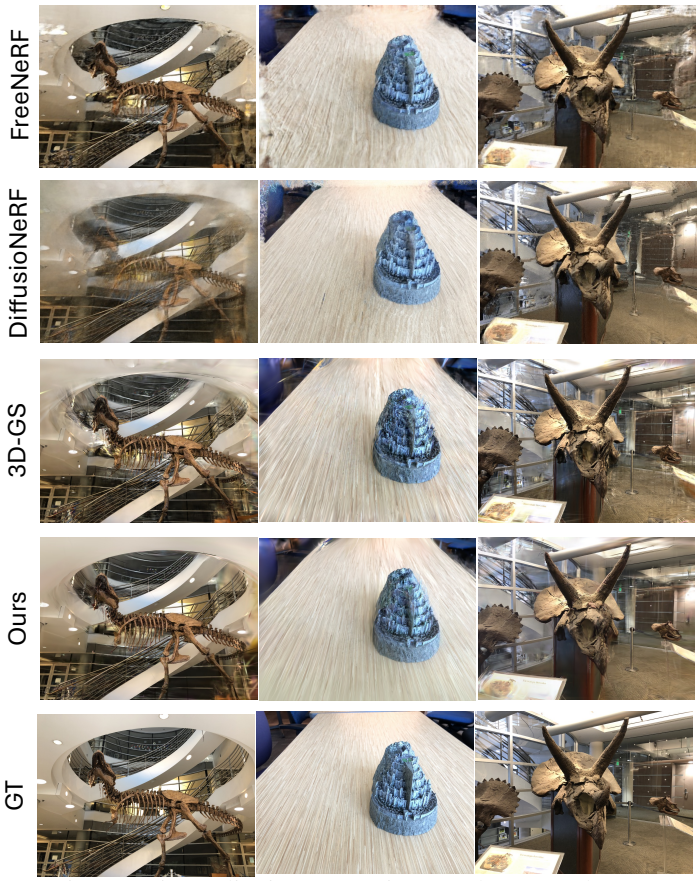


Figure S-5: Qualitative comparison on the LLFF dataset.

Figure S-6: Qualitative comparison with other cutting-edge counterparts on DL3DV dataset with 9 views as input.

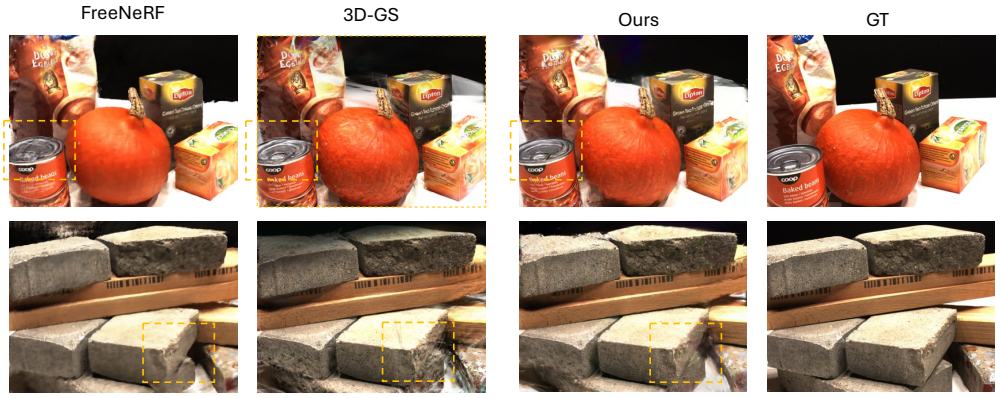


Figure S-7: Qualitative comparison with other methods on DTU dataset. Zoom in for better vision.