NPM-GS: 3D-NATIVE PATCHMATCH FOR SPARSE-VIEW GAUSSIAN SPLATTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Gaussian Splatting achieves high-quality novel-view synthesis with dense observations but degrades severely in sparse-view settings, producing floaters, distorted geometry, and strong train-view overfitting. Existing depth priors, regularization, and feed-forward initialization alleviate these failures, yet residual geometric errors can still survive subsequent Gaussian optimization. We identify a key cause: under sparse observations, geometry can co-adapt with appearance, opacity, scale, and densification, allowing different 3D configurations to explain the same training images. In contrast, cross-view matching directly tests whether a geometric hypothesis is supported by multiple observed views. We further find that this signal is highly geometry-sensitive but effective only within a finite capture range, motivating a bounded discrete search rather than unconstrained continuous optimization. Based on these observations, we propose NPM-GS, a 3D-native hypothesis search and verification framework that directly refines Gaussian centers using discrete cross-view matching. A learned geometric prior defines the search region, while multi-view and spatial agreement identify reliable Gaussians whose geometry is propagated to ambiguous regions through plane-snap and superpixel-based propagation. A subsequent reliability-aware training stage refines appearance while preserving the verified structure. NPM-GS consistently improves sparse-view reconstruction on DTU and Mip-NeRF 360 across PSNR, SSIM, and LPIPS.

1 INTRODUCTION

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) achieves high-quality novel-view synthesis with dense observations, but degrades substantially under sparse views, producing floaters, distorted surfaces, and severe train-view overfitting. Existing methods mitigate these failures through depth or structural priors (Xiong et al., 2024; Zheng et al., 2025), stochastic regularization (Park et al., 2025; Zhang et al., 2024b), or feed-forward geometric initialization (Lin et al., 2025; Wang et al., 2025). While effective, residual geometric errors can still survive subsequent per-scene Gaussian optimization.

Sparse observations, however, do not necessarily lack useful geometric constraints. Classical multi-view stereo, particularly PatchMatch-based methods, explicitly proposes and verifies geometric hypotheses across views (Barnes et al., 2009; Bleyer et al., 2011; Galliani et al., 2015; Schönberger et al., 2016). This contrast suggests that sparse-view failure depends not only on the available image evidence, but also on how geometry is optimized. PatchMatch restricts its search to explicit geometric hypotheses, whereas Gaussian Splatting jointly optimizes geometry, appearance, opacity, scale, and densification through a rendering objective.

We therefore revisit sparse-view Gaussian Splatting from the perspective of its geometric optimization space. Under sparse observations, geometric changes can be compensated by appearance and other Gaussian attributes, allowing substantially different 3D configurations to explain the same training images. Consequently, reducing rendering loss does not necessarily improve geometry, and stronger initialization or depth supervision alone may still leave residual errors after joint optimization.

Cross-view matching provides a more geometry-sensitive signal by directly testing whether a candidate geometry is supported by multiple observed views. However, our analysis shows that this signal is also local: the NCC response forms a clear matching basin near the reference geometry

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

but becomes nearly flat farther away. This motivates combining a learned geometric prior, which provides initialization and a bounded search region, with discrete cross-view verification that selects among candidate geometries.

Based on this observation, we propose NPM-GS, a 3D-native hypothesis search and verification framework for sparse-view Gaussian Splatting. NPM-GS treats each Gaussian center as a geometric hypothesis, searches candidate positions along the viewing ray, and verifies them using multi-view NCC while excluding appearance from the search. Reliable Gaussians are identified through cross-view and spatial agreement, and their geometry is propagated to ambiguous regions through plane-snap and superpixel-based propagation. A subsequent reliability-aware training stage refines appearance while retaining the verified geometric structure. By performing verification directly in the Gaussian representation, NPM-GS transfers cross-view geometric evidence to the primitives ultimately used for rendering.

Our contributions are:

- We diagnose sparse-view Gaussian Splatting from a geometric optimization perspective, showing that its joint rendering objective permits geometry to co-adapt with appearance and other Gaussian attributes, leaving a weakly constrained geometric search space under sparse observations.
- We introduce a 3D-native discrete verification strategy in which a learned geometric prior defines a bounded search region, while cross-view image evidence directly selects Gaussian geometry in the final primitive representation.
- We combine discrete verification with reliability-aware structural propagation and subsequent Gaussian optimization, achieving strong sparse-view novel-view synthesis performance on DTU and Mip-NeRF 360.

2 Related Work

2.1 Sparse-view GS with Additional Priors

A line of work introduces monocular or stereo depth priors as additional supervision (Chung et al., 2024; Li et al., 2024; Xiong et al., 2024; Zhu et al., 2024). Depth estimates or sparse SfM points constrain Gaussian geometry through direct depth losses, normalized depth objectives, or densification guidance. Other methods use cross-view correspondences, optical flow, epipolar constraints, or structural priors to regularize sparse-view reconstruction (Paliwal et al., 2025; Peng et al., 2024; Zheng et al., 2025; Han et al., 2024). Stochastic regularization methods further reduce co-adaptation through Gaussian dropout or consistency-based filtering (Chen et al., 2025a; Park et al., 2025; Zhang et al., 2024b).

These methods reduce sparse-view artifacts by constraining the joint optimization, whereas our focus is complementary: we explicitly separate geometric hypothesis verification from appearance optimization.

2.2 Feed-forward Geometry Priors

Feed-forward reconstruction models estimate geometry from sparse images through explicit epipolar matching, cost volumes, or learned multi-view aggregation (Charatan et al., 2024; Chen et al., 2024; 2025b; Ye et al., 2024; Zhang et al., 2024c;a). More recent large 3D foundation models predict dense depth, point maps, or camera geometry directly from images (Lin et al., 2025; Wang et al., 2025; Leroy et al., 2024; Wang et al., 2024; Zhang et al., 2026), and can provide strong initialization for per-scene Gaussian optimization (Fan et al., 2025).

NPM-GS uses such feed-forward geometry only to initialize a local search region. The final geometric decision is made by cross-view verification from the input images rather than by directly fitting the predicted depth.

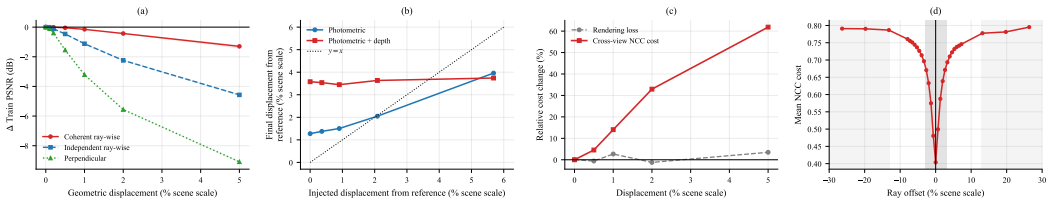

Figure 1: Diagnosing geometric optimization under sparse views on DTU Scan 105 (3 views). (a) Train-view sensitivity to geometric perturbations in different directions. (b) Final displacement from the reference reconstruction after full retraining, with and without depth supervision. (c) Relative sensitivity of the rendering loss and cross-view NCC cost to geometric perturbations. (d) Cross-view NCC landscape along the viewing ray, showing a localized matching basin and a near-flat response at larger offsets.

2.3 PatchMatch Stereo and Structural Propagation

PatchMatch was originally introduced as an efficient randomized correspondence search algorithm Barnes et al. (2009), and was later adapted to stereo reconstruction by optimizing local disparity-plane hypotheses through iterative sampling and propagation Bleyer et al. (2011). This hypothesis-search paradigm was subsequently extended to multi-view stereo, where depth and surface-normal hypotheses are evaluated by cross-view photometric consistency and propagated between neighboring pixels Galliani et al. (2015); Schönberger et al. (2016); Xu & Tao (2019). These methods establish a general search–verify–propagate strategy for geometric reconstruction.

NPM-GS follows this principle but changes the representation on which it operates. Classical PatchMatch stereo verifies per-view depth or plane hypotheses that are later fused into 3D, whereas NPM-GS directly searches and verifies the Gaussian primitives that are subsequently optimized. Our structural propagation is also related to prior work that extends reliable geometric estimates into weakly constrained regions Chaurasia et al. (2013); in our case, propagation is anchored by cross-view verified Gaussians and performed directly in the Gaussian representation.

3 Why Sparse-View 3DGS Fails Geometrically

3D Gaussian Splatting often suffers from floaters, distorted surfaces, and severe training-view overfitting under sparse-view settings. We argue that the issue is not merely insufficient optimization: the Gaussian parameterization leaves a geometric optimization space that is too broad for the available training views to constrain.

Consider a camera center c and a Gaussian parameterized by (μ, Σ). Moving the Gaussian along its viewing ray while scaling its covariance as $\mu_\lambda = c + \lambda(\mu - c)$ and $\Sigma_\lambda = \lambda^2\Sigma$ can preserve its projected center and 2D footprint. As long as visibility ordering is unchanged, the single-view rendering can remain approximately unchanged. Multiple views reduce this ambiguity, but under sparse coverage, limited parallax and overlap still leave substantial freedom in position, scale, opacity, appearance, and densification. Fig. 1(a) empirically supports this anisotropy: spatially coherent perturbations along viewing rays produce substantially smaller training-view degradation than perturbations perpendicular to the rays. The full derivation and additional perturbation results are provided in Appendix A.

We examine this behavior through controlled experiments on DTU Scan 105 with three training views. Displacements are measured relative to a reference reconstruction, rather than ground-truth geometry. As shown in Fig. 1(b), photometric retraining does not consistently return the perturbed reconstruction to this reference configuration. Even without an injected perturbation, retraining produces a reference-relative displacement of 1.27% of the scene scale while increasing training PSNR by 14.18 dB. Thus, improved training-view fit can coexist with substantial changes in geometry. Geometry-first staged optimization also remains less effective than NPM-GS in our tested setting; detailed comparisons are provided in Appendix E.

Adding depth supervision substantially reduces the dependence of the measured final displacement on the injected perturbation (Fig. 1(b)), consistent with a stronger prior constraint. However, similar

reference-relative displacements do not establish convergence to identical geometry or determine absolute accuracy. A depth loss constrains deviations from its target, but does not by itself verify that target against the observed images.

We instead consider cross-view consistency computed directly from the input images. For each candidate geometry, a local patch is projected into other observed views and evaluated using NCC. Figure 1(c) shows a clear contrast between the two signals: over the same perturbation range, the rendering loss changes only weakly, whereas the cross-view NCC cost increases substantially. Each cost is reported as a percentage change relative to its own unperturbed baseline. Cross-view matching therefore provides a much more geometry-sensitive verifier than the rendering objective.

This signal, however, is informative only locally. Fig. 1(d) shows that the NCC landscape forms a sharp basin around the reference geometry, with most of the cost variation concentrated within roughly 3% of the scene scale, while the response becomes nearly flat beyond approximately 13%. Outside this capture region, the problem is not primarily that the local direction is misleading, but that there is little useful local direction at all. This observation motivates our use of a learned prior to define a bounded search region and discrete hypothesis evaluation within that region, rather than unconstrained continuous optimization over the NCC landscape.

Additional diagnostic results, including short re-optimization experiments, complete numerical results, and multi-scale NCC landscapes, are provided in Appendix A.

4 METHOD

4.1 FROM DIAGNOSIS TO 3D-NATIVE VERIFICATION

A seemingly naive way is to refine the DA3 depth maps with a conventional 2D PatchMatch pipeline and then fuse them into 3D. However, in such a pipeline, the objects being verified are per-view depth hypotheses, whereas the final objects being optimized are 3D Gaussian primitives. PatchMatch-style consistency filtering rejects unsupported depth estimates, and multi-view fusion further merges or removes inconsistent samples. This behavior is desirable for producing a reliable surface reconstruction, but it can leave holes when the result is used to initialize 3DGS. Subsequent densification must then introduce new Gaussian primitives that have never undergone the same cross-view verification, reopening the geometry-appearance ambiguity identified in Sec. 3.

We therefore move both hypothesis search and verification into the Gaussian representation itself. Starting from a dense feed-forward initialization, each Gaussian center is directly treated as a geometric hypothesis. Candidate centers are searched along the reference ray, verified against the observed views, and assigned a reliability state before geometric information is propagated to ambiguous regions. The same primitive is therefore searched, verified, propagated, and later optimized. We refer to this representation-consistent design as *3D-native* position hypotheses and verification.

In our implementation, we build NPM-GS on top of 2DGS (Huang et al., 2024), whose explicit normal representation is convenient for tangent-plane patch construction and structural propagation. NPM-GS primarily refines Gaussian centers and is not specific to 2DGS, as confirmed by comparable results with a standard 3DGS implementation shown in Appendix B. An overview of the complete NPM-GS pipeline is shown in Fig. 2.

4.2 INITIALIZATION

We first initialize a dense Gaussian field from Depth Anything 3 (DA3) (Lin et al., 2025). For each training view, every pixel is unprojected using DA3-predicted depth and the camera parameters from dataset to create a Gaussian primitive, yielding dense geometric coverage before refinement. Gaussian color is initialized from the corresponding image pixel, and its local surface orientation is initialized from the normal computed by the predicted depth. We then perform a short warm-up using only RGB and SSIM reconstruction losses, with densification and pruning disabled, so that opacity and other attributes become sufficiently stable for subsequent visibility estimation.

We additionally precompute plane region structure for the propagation stage. Specifically, we combine depth-based superpixels and local planarity cues with SAM2 boundaries (Ravi et al., 2024) to

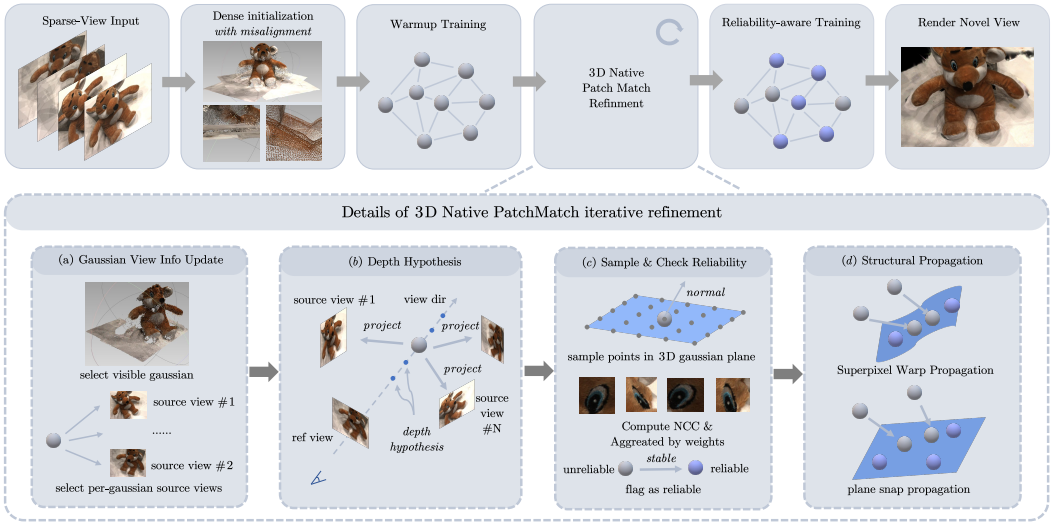

Figure 2: Overview of NPM-GS. Sparse-view images are first converted into a dense Gaussian initialization from DA3 depth (Sec. 4.2). NPM-GS then verifies discrete Gaussian-position hypotheses directly in 3D (Sec. 4.3), identifies reliable primitives, and propagates verified geometry to ambiguous regions (Sec. 4.4). A final reliability-aware stage refines the full Gaussian model (Sec. 4.5).

obtain candidate planar regions while retaining the original superpixel partition for non-planar areas. These regions only define where geometric information can later be propagated.

Finally, NPM-GS explicitly maintains Gaussian visibility across the training views. At each refinement iteration, we render a depth map for every view and compare each Gaussian's projected depth with the rendered depth at the same pixel to determine whether it is visible or occluded. From the visible views, we select the top-K source views according to their viewing direction relative to the Gaussian orientation, while additional geometric reliability is handled by the triangulation and incidence weighting introduced in Sec. 4.3. Visibility and source-view assignments are recomputed as the Gaussian geometry is updated. Exact implementation details are in Appendix C.

4.3 Position Hypotheses and Cross-View Verification

Discrete Position Hypotheses. Our diagnosis in Sec. 3 shows that the most weakly constrained geometric direction is along the viewing ray: moving a Gaussian in depth can substantially change its 3D position while preserving its reference-view projection. We therefore restrict hypothesis generation to this ambiguity direction rather than searching over unconstrained 3D offsets. For a visible Gaussian g in reference view v_r, we freeze its color, opacity, scale, and densification state and refine only its center. Candidate centers are sampled along the reference-view ray through the Gaussian. This converts a free 3D position search into a one-dimensional depth search while keeping the reference-view correspondence fixed. The search also uses a coarse-to-fine schedule: early stages cover a wider depth interval around the current prior estimate and use coarser image resolution, while later stages shrink the interval and evaluate higher-resolution patches.

3D-Native Patch Matching. For each candidate center h, we construct the matching patch directly in 3D space. Specifically, the Gaussian center h together with its current normal n defines a local tangent plane, on which we place a regular grid of 3D sample points centered at h. The sampling interval is chosen such that adjacent samples correspond approximately to a one-pixel displacement in the reference view at the candidate depth, thereby adapting the physical patch spacing to both depth and camera projection.

The same set of 3D sample points is then projected into the reference image and each selected source image using the known camera parameters. We sample the image colors at these projected locations and compute normalized cross correlation (NCC) between the reference and source 2D projected patches. Within each patch, bilateral weights emphasize samples that are both spatially close and photometrically similar to the patch center, following robust patch-matching practice in classical

MVS. In this way, each candidate is evaluated by whether a single local 3D surface hypothesis is consistently supported by multiple observed views.

For hypothesis h and source view k, let$C_{h,k} = 1 - \mathrm{NCC}_{h,k}$. Not every visible source view provides equally useful geometric evidence, so we additionally weight each view by triangulation and incidence. The triangulation weight downweights nearly parallel viewing rays:

$$p_{\text{tri},k} = 1 - \left[\max\left(0,\ \frac{|\cos\theta_{\text{tri},k}| - \cos\theta_{\min}}{1 - \cos\theta_{\min}}\right)\right]^2, \tag{1}$$

where $\theta_{\text{tri},k}$ is the angle between the reference ray and the source-view ray meeting at the candidate point, and $\theta_{\min}$ is a minimum useful triangulation angle. The incidence weight suppresses highly oblique observations,

$$p_{\text{inc},k} = \exp\left(-\frac{[1 - \max(\cos\theta_{\text{inc},k}, 0)]^2}{2\sigma_{\text{inc}}^2}\right), \tag{2}$$

where $\theta_{\text{inc},k}$ is the angle between the source viewing direction and the Gaussian normal. The final candidate cost is

$$C_h = \frac{\sum_{k\in\mathcal{S}} C_{h,k}\, w_k}{\sum_{k\in\mathcal{S}} w_k}, \qquad w_k = p_{\text{tri},k} p_{\text{inc},k}, \tag{3}$$

with $\mathcal{S}$ the selected source-view set. We update the Gaussian center to the candidate with the smallest aggregated cost.

Reliability identification. The minimum NCC candidate is not automatically accepted as reliable. We only label a Gaussian reliable when three independent conditions are satisfied at the finest pyramid level. First, *cross-view consensus*: if each source view selects its own minimum cost depth independently, these argmins must agree within a small geometric tolerance. Second, *spatial consensus*: nearby Gaussians must propose compatible position updates, measured by the variance of the local update field. Third, the Gaussian must be supported by at least two valid source views so that its position is genuinely multi-view constrained. Once all conditions hold, the Gaussian is added to the reliable set $\mathcal{R}$ and its position is frozen for the rest of the NPM refinement. This prevents already verified geometry from drifting during later iterations.

Exact implementation details are in Appendix C.

4.4 Structural Propagation

Cross-view NCC cannot verify every Gaussian. Textureless regions can produce broad, ambiguous matching minima, and non-Lambertian effects such as specular highlights can cause otherwise correct geometry to disagree photometrically across views. Rather than forcing such Gaussians to a low-confidence match, NPM-GS leaves them unreliable and propagates geometry from the verified subset. We use two complementary mechanisms on disjoint regions.

Plane-snap Propagation. For candidate planar regions, we first establish cross-view plane groups. Each planar mask is reprojected to the other training views using the current depth prior; region pairs with sufficient projected overlap are merged with Union-Find into one cross-view group. The resulting group identity is the *plane label*. We then collect all reliable Gaussians whose projections fall inside any region of the group and fit a plane to their 3D centers using RANSAC followed by least-squares refinement on the inliers. Every unreliable Gaussian assigned to the group is projected onto the fitted plane along the ray, and its normal is aligned with the fitted plane normal. This directly fills planar areas in which NCC itself may be nearly uninformative.

Superpixel-warp Propagation. For unreliable Gaussians outside planar regions, we retain the raw superpixel partition from Sec. 4.2. Within each superpixel, reliable Gaussians provide pairs between the DA3 depth and the verified Gaussian depth. We fit an affine correction $d_{\text{corr}} = a\, d_{\text{DA3}} + b$ to these pairs, so that the prior supplies local depth structure while its scale/shift distortion is corrected from verified geometry. For an unreliable Gaussian, we choose the view that observes it most frontally, project its current center into that view, read the affine-corrected depth of the corresponding superpixel, and back-project that depth to obtain the updated 3D center. If a region contains too few reliable seeds, we do not force a fitted correction and retain the prior-based geometry.

Algorithm 1: 3D-Native Hypothesis Search and Verification

Input: Gaussian field $\mathcal{G}$, training views $\mathcal{V}$, planar-region masks $\mathcal{M}$
Output: Refined $\mathcal{G}$ and reliability labels $\mathcal{R}$

```
Initialize G from DA3 depth and warm up appearance/opacity;
Precompute superpixels and candidate planar regions;
for pyramid level ℓ from coarse to fine do
    for refinement iteration do
        for reference view v_r ∈ V do
            Update Gaussian visibility and source-view sets;
            for each visible and unfrozen Gaussian g do
                Select usable source-view pairs with sufficient patch contrast;
                if at least one usable pair is available then
                    Sample candidate centers {h} along the reference ray;
                    Build and project a tangent-plane patch for each h;
                    Score hypotheses using weighted multi-view NCC;
                    Update g to the minimum-cost hypothesis;
                    if at the finest level, at least two valid source views support g, and the evidence test
                     passes then
                        Mark g reliable and freeze its center;
            Plane-snap unreliable Gaussians in planar regions;
            Superpixel-warp remaining unreliable Gaussians;
return G, R;
```

Both propagation steps are executed during each NPM iteration after discrete verification. As the reliable set grows, newly verified geometry becomes additional support for subsequent propagation, while already reliable Gaussians remain frozen. Exact implementation details are in Appendix C.

Pseudocode. Algo. 1 summarizes one full NPM-GS refinement procedure.

4.5 Reliability-Aware Training

After NPM refinement, we resume standard Gaussian training with all attributes trainable and densification enabled. A key concern is that unconstrained photometric optimization could again move the reconstruction away from the geometry established by cross-view verification. We therefore retain the geometric information and reliability labels produced by NPM and use them to guide the final training stage. Importantly, the depth supervision used here is not the original DA3 prediction. Instead, we construct a refined depth map D_{refined} from the geometry recovered during NPM: planar regions use the planes fitted to reliable Gaussians, while non-planar regions use the superpixel-wise affine-corrected DA3 depth estimated from reliable Gaussian anchors. We penalize the difference between rendered depth and this refined geometry,

$$\mathcal{L}_{\text{depth}} = \sum_{p \in \Omega} \left| D_{\text{render}}(p) - D_{\text{refined}}(p) \right|. \tag{4}$$

The same refined geometry is also used for periodic pruning: in reliable planar regions, Gaussians whose projected depth deviates substantially from D_{refined} are removed, targeting floaters that may remain photometrically plausible despite being inconsistent with the verified surface.

We further apply a cross-view color regularization $\mathcal{L}_{\text{cv}}$ to suppress appearance co-adaptation after geometry has been established. This term is used only in reliable planar regions and only where the rendered color of the same surface point varies strongly across training views. Using the verified geometry to establish correspondence, we compute the mean rendered color across the views in which the point is visible and penalize the deviation of each view from this mean:

$$\mathcal{L}_{\text{cv}} = \frac{1}{|\Omega_{\text{cv}}|} \sum_{p \in \Omega_{\text{cv}}} \left| C_r(p) - \bar{C}_r(p) \right|, \qquad \bar{C}_r(p) = \frac{1}{|\mathcal{V}_p|} \sum_{v \in \mathcal{V}_p} C_{r,v}(p), \tag{5}$$

where Ω_{cv} contains only geometrically reliable pixels with sufficiently large cross-view color variance, and $\mathcal{V}_p$ denotes the views in which the corresponding surface point is visible. Restricting

Table 1: Quantitative comparison on DTU with 3, 6, and 9 input views. All methods are evaluated using fixed visibility masks constructed from the three-view training subset, retaining observable foreground and background regions. Best results are shown in **bold**.

Method	3 Views			6 Views			9 Views		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
SparseGS	19.99	0.7721	0.3713	24.25	0.8228	0.3591	26.12	0.8416	0.3514
DNGaussian	17.23	0.6700	0.3812	21.60	0.7320	0.3267	24.25	0.7917	0.2941
NexusGS	20.88	0.7825	0.3598	25.43	0.7838	0.3106	26.55	0.8032	0.2792
CA-3DGS	21.40	0.8252	0.3157	25.15	0.8501	0.2935	27.10	0.8703	0.2780
FLARE	19.54	0.7362	0.4243	20.11	0.7399	0.4320	20.78	0.7461	0.4369
Ours	**24.84**	**0.8259**	**0.2867**	**26.79**	**0.8574**	**0.2675**	**28.07**	**0.8778**	**0.2558**

Table 2: Quantitative comparison on Mip-NeRF 360. Best results are shown in **bold**.

Method	PSNR↑	SSIM↑	LPIPS↓
SparseGS	19.08	0.572	0.384
DNGaussian	19.25	0.534	0.544
FLARE	21.46	0.536	0.265
NexusGS	22.53	0.662	0.299
CA-3DGS	22.64	0.688	0.342
Ours	**24.07**	**0.741**	**0.254**

Table 3: Ablation studies. "D. loss" denotes depth supervision, "D. Hypo." denotes discrete depth-hypothesis search, and "C2F" denotes the coarse-to-fine search strategy.

Configuration	PSNR↑	SSIM↑	LPIPS↓
DA3 Init	19.765	0.6768	0.3557
DA3 Init + D. loss	19.910	0.6822	0.3516
2D PatchMatch init	16.572	0.7140	0.3563
DA3 Init w. Refined D. loss	20.919	0.7990	0.3024
w. D. Hypo.	21.171	0.7989	0.2963
w. D. Hypo. + Superpixel-warp	21.277	0.7963	0.3083
w. D. Hypo. + Plane-snap	21.333	0.7990	0.2992
Full pipeline w/o C2F	21.545	0.8023	0.2963
Full pipeline	**21.623**	**0.8097**	**0.2957**

this term to verified regions avoids using color consistency to compensate for unresolved geometric errors.

Finally, we apply Gaussian dropout following DropGaussian (Park et al., 2025) to further reduce co-adaptation during final rendering optimization. The complete training objective is

$$\mathcal{L} = \lambda_{\text{rgb}}\mathcal{L}_{\text{rgb}} + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}} + \lambda_{\text{depth}}\mathcal{L}_{\text{depth}} + \lambda_{\text{cv}}\mathcal{L}_{\text{cv}}. \tag{6}$$

Loss weights and implementation parameters are provided in Appendix C.

5 Experiments

5.1 Experimental Setup

We evaluate NPM-GS on DTU (Jensen et al., 2014) and Mip-NeRF 360 (Barron et al., 2022) using PSNR, SSIM, and LPIPS. For DTU, we follow prior sparse-view protocols for training-view selection (Niemeyer et al., 2021; Wang et al., 2023), evaluating with 3, 6, and 9 input views. For Mip-NeRF 360, we use four evenly spaced training views and five held-out test views per scene. Camera intrinsics and extrinsics are taken from the datasets. Our DTU evaluation includes both foreground and background pixels that are visible from the training views. For each test view, we construct a visibility mask using the three-view training subset and keep this mask fixed when evaluating all methods with 3, 6, and 9 input views. Details of mask construction are provided in Appendix C.1. We compare with SparseGS (Xiong et al., 2024), DNGaussian (Li et al., 2024), NexusGS (Zheng et al., 2025), CA-3DGS (Chen et al., 2025a), FLARE (Zhang et al., 2026). Details of implementation are provided in Appendix C.

5.2 Main Results

Table 1 and Table 2 shows that NPM-GS consistently improves over the compared sparse-view methods on both DTU and Mip-NeRF 360. It is worth noting that the performance gap between our method and the competing approaches gradually narrows as the number of input views increases.

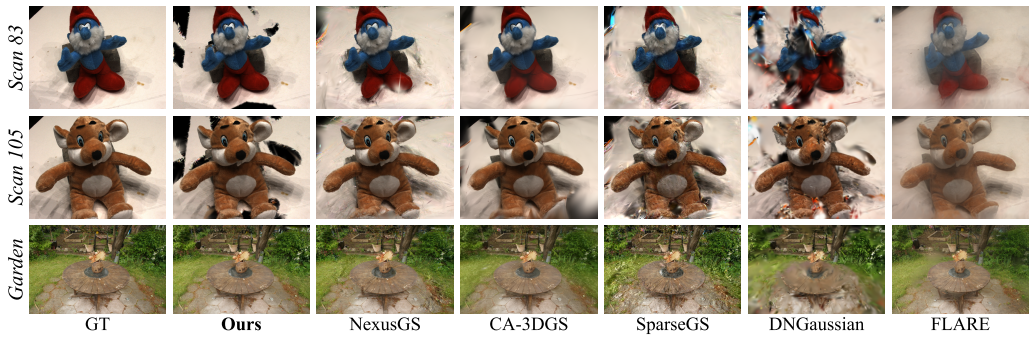

Figure 3: Qualitative comparison on DTU (*Scan 83*, *Scan 105*) and Mip-NeRF 360 (*Garden*). NPM-GS produces the most faithful renderings, with fewer floaters and better detail preservation than all baselines.

This trend is consistent with our earlier analysis in Sec. 3: additional views provide stronger geometric constraints, effectively narrowing the feasible solution space of the search domain. At the same time, with more observations, the estimated optical flow, scene structure, and depth priors become increasingly accurate. Consequently, competing methods also benefit from a better-constrained optimization problem and more reliable geometric cues, leading to improved performance and a reduced relative advantage of our method in denser-view settings. Fig. 3 presents representative qualitative comparisons on DTU Scan 83 and Scan 105 and the Garden scene from Mip-NeRF 360. More qualitative comparisons on DTU and Mip-NeRF 360 are provided in Appendix F. We additionally provide a runtime comparison in Appendix 6, where NPM-GS shows competitive efficiency among optimization-based sparse-view Gaussian splatting methods.

5.3 Component Ablation

Table 3 compares alternative uses of the feed-forward depth prior and the main components of NPM-GS. Directly supervising Gaussian training with either the original DA3 depth or the NPM-refined depth remains less effective than explicitly correcting Gaussian geometry, indicating that the gain cannot be explained by stronger depth supervision alone. Likewise, using DA3 to initialize a conventional 2D PatchMatch pipeline performs poorly in our setting. The resulting reconstruction contains only 26,554 points, compared with 57,874 points from the COLMAP dense reconstruction, providing substantially weaker geometric support for subsequent Gaussian optimization.

The remaining ablations show complementary contributions from discrete depth-hypothesis search and structural propagation. Plane snapping and superpixel-based propagation extend reliable corrections into regions where direct matching is ambiguous, while coarse-to-fine search provides an additional refinement of the hypothesis search. Overall, the results support NPM-GS as a combination of discrete cross-view verification and structure-aware propagation rather than a stronger depth-supervision scheme alone.

6 Conclusion

We presented NPM-GS for sparse-view Gaussian Splatting. Our diagnosis shows that sparse-view failure arises from a geometric search space that is too broad for rendering supervision to constrain reliably, while cross-view verification is informative only within a finite capture region. NPM-GS therefore uses learned depth to define a bounded search region and discrete cross-view verification to select geometry directly on Gaussian primitives.

Our method assumes static scenes and does not explicitly handle dynamic objects. Moreover, NCC-based cross-view matching relies on local texture; while textureless planar regions can often be recovered through structural propagation, more general low-texture regions remain challenging. Future work will explore feature-based matching and test-time optimization of feed-forward reconstruction outputs to improve robustness in these cases.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript organization and language refinement, as well as code assistance and debugging. All AI-assisted code and text were reviewed and verified by the authors. The authors take full responsibility for the final content of the paper, including all implementations, experiments, results, and scientific claims.

ETHICS STATEMENT

This work studies sparse-view 3D reconstruction using existing public benchmark datasets and does not involve human-subject experiments or the collection of personal data. We are not aware of specific ethical concerns arising from the methodology beyond the general considerations associated with image-based 3D reconstruction.

REPRODUCIBILITY STATEMENT

We provide the algorithmic details of NPM-GS in the main paper and Appendix, including the complete refinement procedure, implementation parameters, dataset splits and evaluation protocols. The Appendix additionally reports component ablations, alternative optimization strategies, runtime comparisons, and implementation details for both the 2DGS- and 3DGS-based variants. Code and evaluation scripts will be released upon publication to facilitate reproduction of the reported results.

REFERENCES

Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics*, 28 (3):1–11, July 2009. ISSN 0730-0301, 1557-7368. doi: 10.1145/1531326.1531330.

Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields, March 2022.

Michael Bleyer, Christoph Rhemann, and Carsten Rother. PatchMatch Stereo - Stereo Matching with Slanted Support Windows. In *Procedings of the British Machine Vision Conference 2011*, pp. 14.1–14.11, Dundee, 2011. British Machine Vision Association. ISBN 978-1-901725-43-8. doi: 10.5244/C.25.14.

David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D Gaussian Splats from Image Pairs for Scalable Generalizable 3D Reconstruction, April 2024.

Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics*, 32(3):1–12, June 2013. ISSN 0730-0301, 1557-7368. doi: 10.1145/2487228.2487238.

Kangjie Chen, Yingji Zhong, Zhihao Li, Jiaqi Lin, Youyu Chen, Minghan Qin, and Haoqian Wang. Quantifying and Alleviating Co-Adaptation in Sparse-View 3D Gaussian Splatting, September 2025a.

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, November 2024.

Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. MVSplat: Efficient 3D Gaussian Splatting from Sparse Multi-View Images. volume 15079, pp. 370–386. 2025b. doi: 10.1007/978-3-031-72664-4_21.

Jaeyoung Chung, Jeongtaek Oh, and Kyoung Mu Lee. Depth-Regularized Optimization for 3D Gaussian Splatting in Few-Shot Images, January 2024.

Zhiwen Fan, Wenyan Cong, Kairun Wen, Kevin Wang, Jian Zhang, Xinghao Ding, Danfei Xu, Boris Ivanovic, Marco Pavone, Georgios Pavlakos, Zhangyang Wang, and Yue Wang. InstantSplat: Sparse-view Gaussian Splatting in Seconds, July 2025.

Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively Parallel Multiview Stereopsis by Surface Normal Diffusion. In *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pp. 873–881, Santiago, Chile, December 2015. IEEE. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.106.

Liang Han, Junsheng Zhou, Yu-Shen Liu, and Zhizhong Han. Binocular-Guided 3D Gaussian Splatting with View Consistency for Sparse View Synthesis, October 2024.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2D Gaussian Splatting for Geometrically Accurate Radiance Fields, June 2024.

Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanaes. Large Scale Multi-view Stereopsis Evaluation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 406–413, Columbus, OH, USA, June 2014. IEEE. ISBN 978-1-4799-5118-5. doi: 10.1109/CVPR.2014.59.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Trans. Graph.*, 42(4):1–14, August 2023. ISSN 0730-0301, 1557-7368. doi: 10.1145/3592433.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding Image Matching in 3D with MASt3R, June 2024.

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, March 2024.

Haotong Lin, Sili Chen, Junhao Liew, Donny Y. Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the Visual Space from Any Views, November 2025.

Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs, December 2021.

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. volume 15088, pp. 19–37. 2025. doi: 10.1007/978-3-031-73404-5_2.

Hyunwoo Park, Gun Ryu, and Wonjun Kim. DropGaussian: Structural Regularization for Sparse-view Gaussian Splatting, April 2025.

Rui Peng, Wangze Xu, Luyang Tang, Liwei Liao, Jianbo Jiao, and Ronggang Wang. Structure Consistent Gaussian Splatting with Matching Prior for Few-shot Novel View Synthesis, November 2024.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment Anything in Images and Videos, October 2024.

Johannes L. Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise View Selection for Unstructured Multi-View Stereo. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (eds.), *Proc. Eur. Conf. Comput. Vis. (ECCV)*, volume 9907, pp. 501–518, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46486-2 978-3-319-46487-9. doi: 10.1007/978-3-319-46487-9_31.

Guangcong Wang, Zhaoxi Chen, Chen Change Loy, and Ziwei Liu. SparseNeRF: Distilling Depth Ranking for Few-shot Novel View Synthesis, August 2023.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual Geometry Grounded Transformer, March 2025.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D Vision Made Easy, December 2024.

Haolin Xiong, Sairisheek Muttukuru, Rishi Upadhyay, Pradyumna Chari, and Achuta Kadambi. SparseGS: Real-Time 360{\deg} Sparse View Synthesis using Gaussian Splatting, May 2024.

Qingshan Xu and Wenbing Tao. Multi-Scale Geometric Consistency Guided Multi-View Stereo. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 5478–5487, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-7281-3293-8. doi: 10.1109/CVPR.2019.00563.

Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No Pose, No Problem: Surprisingly Simple 3D Gaussian Splats from Sparse Unposed Images, October 2024.

Chuanrui Zhang, Yingshuang Zou, Zhuoling Li, Minmin Yi, and Haoqian Wang. TranSplat: Generalizable 3D Gaussian Splatting from Sparse Multi-View Images with Transformers, August 2024a.

Jiawei Zhang, Jiahe Li, Xiaohan Yu, Lei Huang, Lin Gu, Jin Zheng, and Xiao Bai. CoR-GS: Sparse-View 3D Gaussian Splatting via Co-Regularization, July 2024b.

Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. GS-LRM: Large Reconstruction Model for 3D Gaussian Splatting, April 2024c.

Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. FLARE: Feed-forward Geometry, Appearance and Camera Estimation from Uncalibrated Sparse Views, January 2026.

Yulong Zheng, Zicheng Jiang, Shengfeng He, Yandu Sun, Junyu Dong, Huaidong Zhang, and Yong Du. NexusGS: Sparse View Synthesis with Epipolar Depth Priors in 3D Gaussian Splatting, March 2025.

Zehao Zhu, Zhiwen Fan, Yifan Jiang, and Zhangyang Wang. FSGS: Real-Time Few-shot View Synthesis using Gaussian Splatting, June 2024.

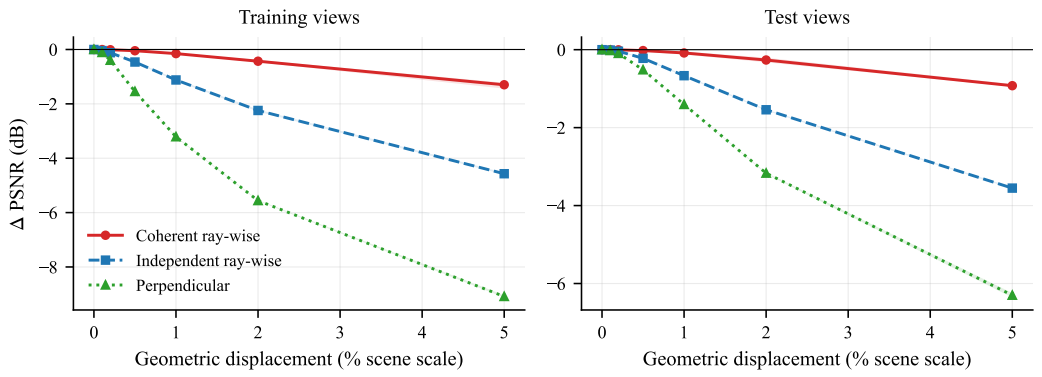

Figure 4: **Directional sensitivity to geometric perturbations.** Train- and test-view PSNR changes under coherent ray-wise, independent ray-wise, and perpendicular Gaussian displacements. Spatially coherent displacement along viewing rays is substantially less visible to the rendering objective than perpendicular motion.

A Additional Diagnostic Analysis

We provide additional controlled experiments supporting the geometric diagnosis in Sec. 3. Unless otherwise specified, all experiments in this section are conducted on DTU Scan 105 using three training views. The experiments start from the same reference reconstruction and perturb its Gaussian geometry by a prescribed fraction of the scene scale. We emphasize that the displacement measurements below are relative to this reference reconstruction rather than to ground-truth 3D geometry.

A.1 Directional Sensitivity of the Rendering Objective

Fig. 4 reports the complete directional perturbation experiment corresponding to Fig. 1(a). We compare three perturbation modes: spatially coherent displacement along viewing rays, independent per-Gaussian displacement along viewing rays, and displacement perpendicular to the viewing rays.

The rendering response is strongly anisotropic. Spatially coherent ray-wise perturbations produce substantially smaller degradation than perpendicular perturbations of the same magnitude. At a displacement of 5% of the scene scale, the train-view PSNR decreases by only 1.30 dB for the coherent ray-wise perturbation, compared with 9.09 dB for the perpendicular perturbation. Independent ray-wise perturbations lie between these two cases, as they additionally disrupt local surface coherence. The same ordering is also visible on the held-out view. These results support the existence of weakly constrained geometric directions associated with coherent motion along viewing rays.

A.2 Short Re-optimization after Geometric Perturbation

We next test whether standard photometric optimization can immediately undo the injected geometric perturbations. Fig. 5 compares the rendering quality before and after a short re-optimization stage.

Photometric optimization rapidly recovers the training views even after large geometric perturbations. For example, at 5% displacement, the train-view PSNR increases from 21.38 dB to 34.85 dB after re-optimization. In contrast, the held-out-view PSNR improves only from 19.39 dB to 21.03 dB and remains substantially below the unperturbed result. This discrepancy indicates that restoring the training images does not necessarily restore the underlying geometry. At small perturbations, the ambiguity is even more apparent: a 0.5% perturbation produces essentially no degradation after re-optimization on the held-out view.

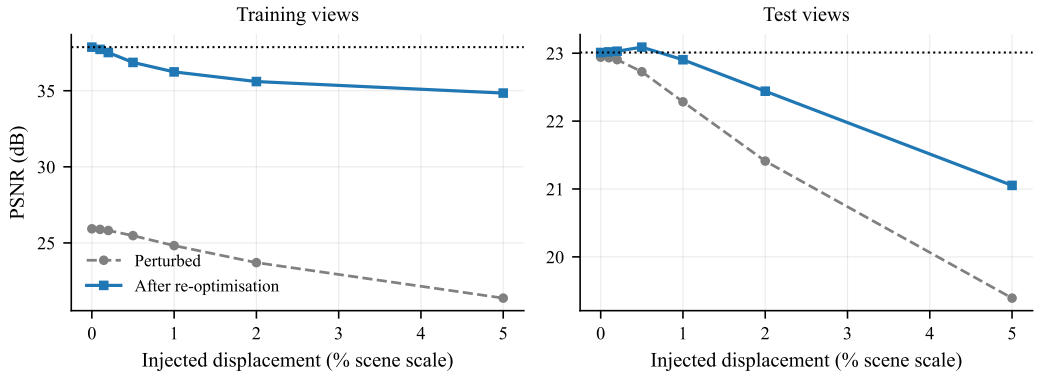

Figure 5: **Short photometric re-optimization after geometric perturbation.** Training views can be re-fitted effectively after perturbation, while the held-out view retains substantially larger degradation as the injected displacement increases. The dotted line indicates the re-optimized result without injected displacement.

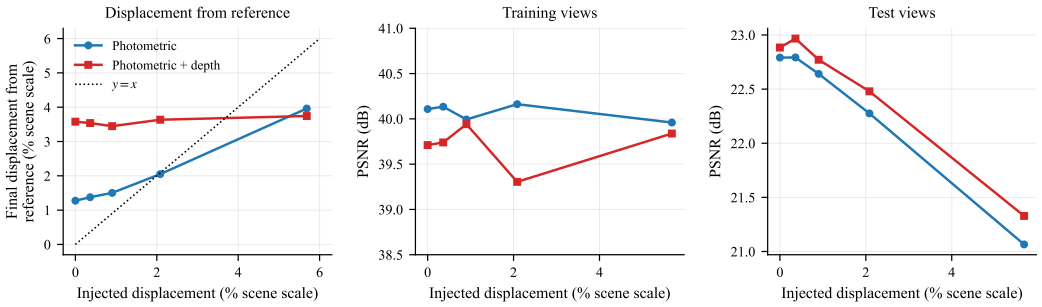

Figure 6: **Retraining after geometric perturbation.** Left: final displacement relative to the reference reconstruction. Middle and right: corresponding train- and test-view PSNR. We compare photometric training with and without depth supervision. Reference-relative displacement is not a measure of ground-truth geometric accuracy.

A.3 Full Retraining and the Effect of Depth Supervision

We further repeat the perturbation experiment using full Gaussian retraining, with all Gaussian attributes trainable and densification and pruning enabled. Figure 6 reports the final displacement from the reference reconstruction together with the corresponding train- and test-view PSNR. We compare photometric training alone against training with additional depth supervision.

With photometric training alone, the final reference-relative displacement remains dependent on the injected perturbation. A linear fit has a slope of approximately 0.47. We interpret this slope as a measure of initialization sensitivity, rather than as the fraction of ground-truth geometric error that survives retraining. The nonzero displacement in the unperturbed run further shows that improved training-view fit need not preserve the reference configuration.

Adding depth supervision produces a much narrower range of final reference-relative displacements across the five initializations. This is consistent with the depth term imposing a stronger prior constraint. However, similar scalar displacement values do not imply that the resulting depth maps are identical, nor do they establish geometric accuracy. The experiment therefore characterizes the response to initialization under different objectives, rather than proving convergence to a common correct solution.

Across both variants, training-view PSNR varies relatively little despite differences in reference-relative displacement and held-out rendering quality. This highlights the limited ability of training-view fit alone to distinguish these reconstruction outcomes.

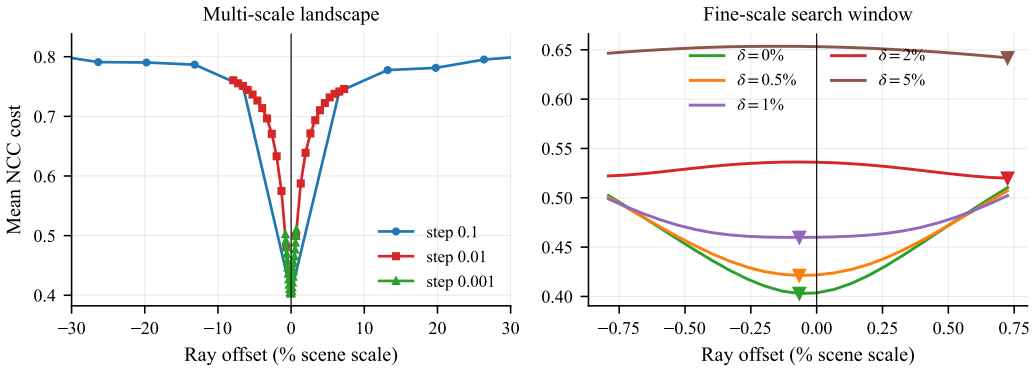

Figure 7: **Multi-scale cross-view NCC landscape.** Left: NCC cost along the viewing ray at coarse, intermediate, and fine search scales. The matching response has a pronounced local basin but becomes nearly flat at large offsets. Right: fine-scale NCC curves under different injected displacements. When the required correction exceeds the fine search window, the minimum approaches the search boundary and the local signal weakens.

A.4 Multi-scale NCC Landscape

Fig. 7 expands the NCC landscape shown in Fig. 1(d). We evaluate the mean cross-view NCC cost by scanning candidate positions along the viewing ray at three search scales. These curves summarize the mean response in this diagnostic setting; individual weakly textured patches may instead exhibit nearly flat costs across the entire search interval.

The coarse-scale curve reveals a deep and smooth basin around the reference geometry, but the cost rapidly saturates away from this region. Most of the cost variation occurs within only a few percent of the scene scale; at larger offsets, further displacement changes the NCC cost only weakly. Consequently, cross-view matching provides a strong local geometric signal but a weak long-range optimization direction.

The fine-scale curves additionally show why the search interval must cover the true correction. When the initial displacement remains within the fine search window, the minimum lies inside the evaluated interval and provides a useful local correction. Once the required correction exceeds the search window, the minimum moves toward the boundary and the cost variation inside the window becomes much weaker. This behavior motivates the coarse-to-fine search used in NPM-GS: coarse stages first bring the geometry into the local matching basin, after which progressively finer stages refine the solution.

A.5 Summary

Together, these controlled experiments support the design choices in Sec. 3. First, rendering-based supervision is highly anisotropic and leaves coherent ray-wise geometric changes weakly constrained. Second, both short and full photometric retraining can recover the observed training images without reliably restoring the reference geometry. Third, depth supervision strongly constrains the geometry but acts primarily as an attractor toward the predicted prior. Finally, cross-view NCC provides a much more geometry-sensitive signal, but only within a finite capture region. These observations motivate combining a feed-forward geometric prior for initialization with bounded, discrete cross-view verification rather than relying solely on continuous rendering-based optimization.

B Implementation of the 3DGS-Based Variant

To isolate the effect of the Gaussian representation, we implement the 3DGS-based variant by replacing only the primitive parameterization and rasterizer, while keeping the remaining NPM-GS pipeline unchanged. In particular, the dataset splits, training procedure and losses, structural propagation, and evaluation are identical to those used in the 2DGS implementation. We also preserve the

Table 4: **Implementation parameters of NPM-GS.** K denotes the number of source views used for cross-view verification, $\theta_{\min}$ the minimum triangulation angle, and σ_{inc} the incidence-angle weighting parameter.

Parameter	Value
Patch size	11×11
Depth hypotheses	20
Source views K	3
Visibility tolerance	1% of depth
Normal-view gate	$\angle(\mathbf{v}, \mathbf{n}) \leq 75^\circ$
Minimum triangulation angle $\theta_{\min}$	1°
Incidence weighting σ_{inc}	0.5
Bilateral $\sigma_{\text{spatial}}/\sigma_{\text{color}}$	$3.0/0.3$
Spatial-consensus neighbors	10 (3D KNN)
Reliability threshold (cross-view / spatial)	0.1% of Scene Scale
Plane-snap RANSAC	128 iterations
Plane-snap inlier threshold	0.2% of Scene Scale
Minimum reliable plane seeds	10
Cross-view plane merging	Union-Find, overlap $> 30\%$
Depth-search schedule	$1 \times r = 0.1,\ 1 \times r = 0.01,\ 3 \times r = 0.001 \times SceneScale$
Gaussian dropout	$0.1 \rightarrow 0.3$ linearly
Training iterations (DTU)	4,500
Training iterations (Mip-NeRF 360)	9,500

same renderer interface so that all downstream training and evaluation code can be reused without modification.

The 2D surfel primitive is replaced by a standard 3D Gaussian with three scaling dimensions and a standard 3D covariance parameterization. The Gaussian rotation continues to encode the depth-derived local orientation, and we use its third rotation axis as the local surface normal, following the same convention used during initialization. Consequently, the NPM search, source-view selection, tangent-plane construction, and propagation stages require no changes.

We replace the 2DGS rasterizer with a standard 3DGS rasterizer. Since the rasterizer outputs only three-channel colors, we can treat the Gaussian normals as colors to obtain the rendered normal map. Expected surface depth is computed by normalizing accumulated depth by opacity, and surface normals are additionally recovered from this depth map, matching the quantities expected by the original NPM-GS pipeline. Renderer outputs retain the same keys as the 2DGS implementation, allowing the training, propagation, and evaluation modules to remain unchanged.

Thus, the 3DGS experiment changes only the underlying Gaussian representation and rasterization process. All NPM-specific components and evaluation protocols are held fixed, providing a controlled test of whether the proposed search-and-verification framework depends on the 2DGS formulation.

C Implementation Details and Hyperparameters

Implementation Details. Table 4 summarizes the main parameters used in NPM-GS. These parameters control the local PatchMatch search, cross-view verification, reliability estimation, structural propagation, and the subsequent Gaussian optimization.

Loss Parameters. The loss weights used during the final training stage are summarized in Table 5. We follow the default pruning schedule of the underlying Gaussian implementation.

C.1 Evaluation Masks

To evaluate both foreground and background reconstruction, we use test-view visibility masks rather than restricting evaluation to foreground regions. We first obtain reference depth maps through COLMAP dense reconstruction and manually complete textureless regions using fitted plane equations where applicable. These reconstructed reference depths are used to determine the evaluation

Table 5: **Loss weights for reliability-aware training.** The depth loss is enabled after 500 iterations, while the cross-view loss is applied only within planar regions.

Loss term	Weight	Activation
$\lambda_{\text{rgb}}\mathcal{L}_{\text{rgb}}$	0.8	All iterations
$\lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}}$	0.2	All iterations
$\lambda_{\text{depth}}\mathcal{L}_{\text{depth}}$	0.05	Iteration > 500
$\lambda_{\text{cv}}\mathcal{L}_{\text{cv}}$	0.1	Planar regions only

Table 6: Per-scene runtime comparison. Runtime is measured using the implementations employed in our experiments. Feed-forward methods are substantially faster, while NPM-GS remains comparable in runtime to optimization-based sparse-view Gaussian splatting methods.

Method	Runtime
NexusGS	$\sim$5.5 min
SparseGS	$\sim$10 min
DNGaussian	$\sim$12.5 min
NPM-GS (Ours)	$\sim$13.5 min
CA-3DGS	$\sim$32 min
FLARE	$\sim$4.5 sec

region. Specifically, we back-project the reference depths from the training subset into 3D and reproject the resulting points into each test view. The resulting mask identifies test-view pixels observable from at least one of these training views. For each test view, the same mask is used for all methods and all three input settings (3, 6, and 9 views) on DTU, as well as MipNeRF 360. Thus, increasing the number of training views does not change the evaluated image region.

D Runtime Comparisons on DTU

Table 6 reports per-scene computation time on DTU with three input views, measured on a workstation with an Intel Core i7-14700KF CPU and an NVIDIA RTX 5090 GPU. The reported timings exclude auxiliary prior preprocessing, such as DA3 inference, and should not be interpreted as end-to-end runtimes. NPM-GS takes approximately 13.5 minutes per scan under this timing convention. As expected, feed-forward methods such as FLARE are substantially faster.

E Direct Optimization Ablation

We further investigate whether the failure of direct optimization can be alleviated by decoupling geometry and appearance. All variants use the same DA3 dense initialization. We denote Gaussian geometry parameters (position, scale, and rotation) by G, opacity by O, color/SH coefficients by C, and refined-depth supervision by D.

The results show that simply separating geometry and appearance optimization does not resolve the underlying ambiguity. Optimizing position alone is highly unstable, while allowing opacity to adapt substantially improves the best intermediate solution. However, the degradation from the best to the final checkpoint indicates that continued continuous optimization can still move the reconstruction away from a better geometric configuration.

More importantly, a staged strategy that first optimizes geometry and opacity and subsequently releases the appearance parameters does not recover the lost performance. Adding refined-depth supervision produces only a minor change. This suggests that freezing appearance during the early stage does not, by itself, make the geometry optimization sufficiently constrained: once the geometry is driven toward an incorrect solution, the subsequent appearance stage cannot reliably correct it.

Allowing additional geometric degrees of freedom through scale and rotation produces a stronger direct-optimization baseline, but the same behavior remains: the best intermediate checkpoint is better than the final result and still falls below NPM-GS. These experiments indicate that the limitation

Table 7: Ablation of direct and staged Gaussian optimization on **DTU Scan 105 with four input views**. G denotes geometry (position, scale, and rotation), O opacity, C color/SH coefficients, and D refined-depth supervision. "Best" and "final" report the best checkpoint and the final optimization result, respectively.

Configuration	PSNR↑	SSIM↑	LPIPS↓
xyz	12.27	0.677	0.510
xyz + O (best)	22.05	0.770	0.428
xyz + O (final)	21.09	0.736	0.450
xyz + $O \rightarrow C + O$	19.86	0.695	0.457
xyz + $O + D \rightarrow C + O + D$	19.97	0.706	0.449
$G + O + D \rightarrow C + O + D$ (best)	22.76	0.753	0.390
$G + O + D \rightarrow C + O + D$ (final)	21.74	0.753	0.390
NPM-GS	**23.19**	**0.824**	**0.357**

is not simply caused by joint optimization of geometry and appearance. Rather, continuous losses alone provide insufficient geometric discrimination in ambiguous regions. NPM-GS addresses this issue by explicitly evaluating discrete geometric hypotheses across views and retaining only geometrically reliable updates before subsequent appearance optimization.

F Additional Validation Results

We provide additional qualitative results on the DTU and Mip-NeRF 360 datasets to examine whether the observations reported in the main paper generalize beyond the primary evaluation setting. For both datasets, we follow exactly the same evaluation protocol described in the main paper. No dataset-specific modification is introduced. We provide the rest results of DTU and Mip-NeRF 360 datasets in Fig. 9 and 8, respectively.

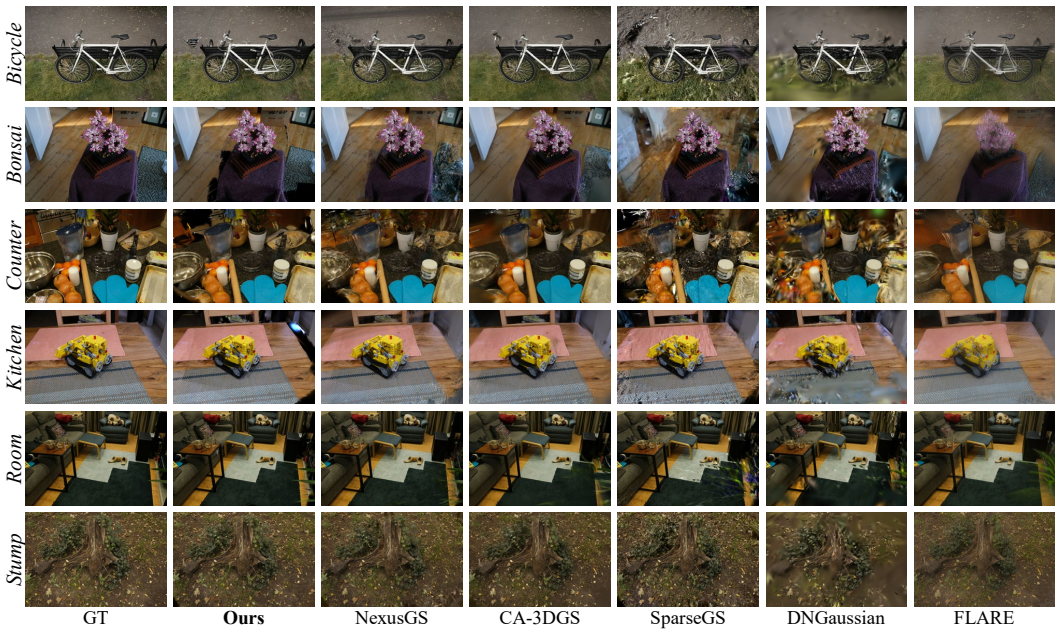

Figure 8: Additional results on Mip-NeRF 360. Full validation results on the Mip-NeRF 360 dataset under the same experimental protocol used in the main paper. The results further confirm that the observations reported in the main paper persist in large-scale real-world scenes.



Figure 9: Additional results on DTU. Full validation results on the DTU dataset under the same experimental protocol used in the main paper. These results complement the representative examples in main paper and show that the observed behavior remains consistent across the evaluated DTU scenes.