

Exploration Matters for Escaping the Blur Trap in 3D Gaussian Splatting

Chengbo Wang¹, Guozheng Ma², Jinhong Wu¹, Tie Ji¹, Yizhen Lao¹

¹Hunan University, ²Nanyang Technological University

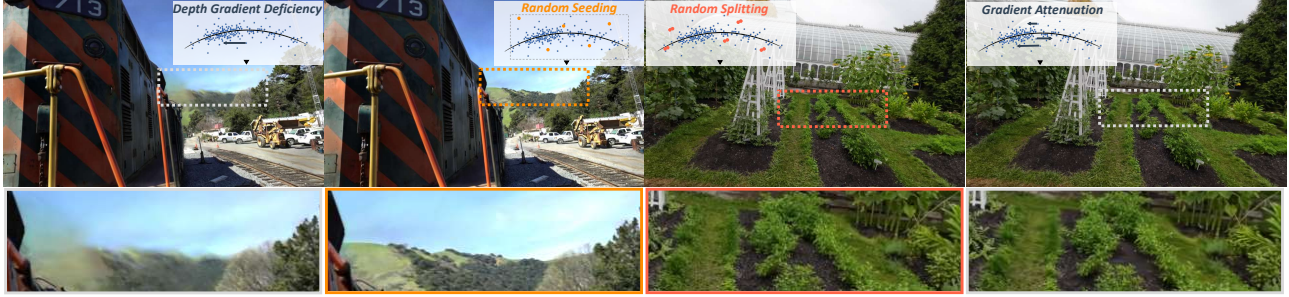


Figure 1: Escaping the Blur Trap via Stochastic Exploration. We identify the *Blur Trap* as a fundamental limitation intrinsic to 3DGS, rooted in an inherent exploitation-only optimization bias. It manifests as two variants: the Far-Side Blur Trap (Left) triggered by depth-gradient deficiency and the Near-Side Blur Trap (Right) driven by 2D-gradient attenuation. To break this deadlock, we introduce minimal exploration operators: *Random Seeding* and *Random Splitting*. These straightforward interventions bypass gradient-dependent optimization, consistently recovering structural details and proving that simple explicit exploration is highly effective for escaping the Blur Trap without complex modifications.

Abstract

3D Gaussian Splatting (3DGS) employs Gaussian primitives for explicit scene representation, facilitating real-time, high-fidelity reconstruction and novel view synthesis of complex scenes. However, the explicit modeling inherent in 3DGS introduces a gradient bias during optimization, rendering its non-convex optimization process highly susceptible to convergence toward local suboptimal solutions. ***This constitutes a fundamental limitation in 3DGS optimization, which we term the Blur Trap.*** To address this limitation, we integrate simple explicit exploration into the 3DGS optimization framework. First, through rigorous mathematical analysis of the 3DGS optimization formulation, we identify the underlying optimization bias responsible for the Blur Trap and categorize it into two distinct subtypes: the Far-Side Blur Trap and the Near-Side Blur Trap. Subsequently, we propose two highly straightforward exploration strategies (*Random Seeding* and *Random Splitting*) to mitigate the far-side and near-side blur traps, respectively. Experimental validation demonstrates that the incorporation of these exploration operators effectively and complementarily overcome the Blur Trap, achieving high-quality rendering performance across multiple datasets. Project page: <https://chengbo-wang.github.io/ExploreGS/>.

1. Introduction

3D Gaussian Splatting (3DGS) [1] has rapidly become a leading approach for novel view synthesis, offering real-time rendering together with photorealistic fidelity through an explicit representation built from anisotropic Gaussian primitives. Unlike implicit neural representations such as NeRF [2], 3DGS owes much of its efficiency to a physically grounded design choice: each Gaussian is updated by gradients derived directly

from the camera projection model, with no intermediate neural network in the loop. This design has driven impressive progress on standard benchmarks. Yet a recurring failure pattern can be observed across 3DGS reconstructions. ***Certain regions in the rendered scene remain visibly blurry no matter how many training views are provided.*** As shown in Figure 1, this is not random noise. It appears consistently in two kinds of regions: distant content such as mountains and skylines, and occluded near-field regions such as the grass

behind the rack. The persistence of these artifacts even under abundant supervision indicates that the cause is not insufficient data but something rooted in the optimization process itself. We refer to this systematic failure as the **Blur Trap**.

To understand why the Blur Trap forms, we trace the issue back to the gradient that drives 3DGS optimization. The 3D position update of each Gaussian is composed of three branches that correspond to the 2D projected position, the 2D covariance, and the spherical harmonic coefficients. Empirically, the 2D position component dominates the other two by two to three orders of magnitude across the entire training process, so *the 3D position update is effectively governed by 2D reprojection error alone*. We then ask what this dominant component looks like geometrically. Through a careful derivation of the camera projection pipeline, we prove that *the 3D position update derived from the 2D gradient is strictly orthogonal to the viewing ray from the camera center to the Gaussian*. The practical consequence is that the optimizer can only move primitives within the plane perpendicular to the viewing ray and never along the depth direction of the viewing ray. Compounding this, the same 2D gradient also serves as the criterion for adaptive densification, and the α -blending pipeline systematically attenuates this signal for occluded Gaussians, since *later-ranked primitives in the depth-sorted sequence receive transmittance-weighted gradients that decay rapidly*. The Blur Trap therefore decomposes into two subtypes. The *Far-Side Blur Trap* arises in distant regions where there is no depth-directed optimization signal to drive Gaussians to their correct depth. The *Near-Side Blur Trap* arises in occluded foreground regions where the densification trigger is suppressed before it can fire.

Existing efforts to improve 3DGS have largely circled around the Blur Trap without naming or resolving it. A first line of work refines the densification criterion. AbsGS [3] addresses gradient direction conflicts among sub-pixel signals, PixelGS [4] reweights gradients by pixel coverage, and similar variants [5, 6] adjust when and how clone or split operations fire. These methods all consume the same input: a gradient signal that already carries the two biases described above. Adjusting the threshold or the weighting does not change the fact that the signal itself lacks depth information and is attenuated by α -blending. A second line of work, exemplified by HoGS [7], reparameterizes Gaussian positions in homogeneous coordinates to better express far-field content. This changes how distant primitives are represented but does not introduce any new optimization signal in the depth direction, so the underlying gradient deficiency persists. A third line of work, most notably 3DGS-MCMC [8], injects Langevin-style noise into Gaussian positions and reinterprets the entire pipeline as MCMC sampling. This is the closest in spirit to our perspective. However, the noise there is applied uniformly across the scene and is not tied to a specific diagnosis of why 3DGS fails. As a result, the method treats all primitives with the same exploratory mechanism, even though, as we show, the failure has two distinct geometric origins that call for two

different remedies.

These observations suggest a simple reframing. 3DGS optimization is fundamentally a high-dimensional non-convex problem, and the broader optimization literature has long recognized that escaping local optima in such landscapes requires some form of exploration, whether through stochastic gradient noise [9, 10], perturbed gradient descent [11], or simulated annealing. Standard neural network training enjoys several natural sources of such exploration, including the noise inherent in mini-batch stochastic gradients and implicit regularization from dropout or weight decay. 3DGS enjoys almost none of these. Training proceeds with a minibatch size of one, the update rule contains no implicit regularization, and as established above, the physically derived gradient is itself structurally biased in ways that prevent the optimizer from probing either depth or occluded regions. From this vantage point, the Blur Trap is not a surprise but the predictable outcome of running a deterministic, gradient-dominated optimizer on a landscape whose dominant signal cannot point in the directions that matter. The natural remedy is to put exploration back in, and to design it so that it directly compensates for what the gradient cannot supply.

Guided by this principle, we introduce two minimal exploration operators, each tied to one subtype of the Blur Trap. **Random Seeding** addresses the Far-Side Blur Trap by periodically injecting candidate Gaussians at randomly sampled 3D positions across the scene. This bypasses the orthogonality constraint, since new primitives can land at depths that no existing Gaussian could have reached through gradient updates alone. Seeds that fall in geometrically invalid regions are removed by standard opacity pruning, and seeds that land in plausible regions are picked up by the regular optimization loop and refined. **Random Splitting** addresses the Near-Side Blur Trap by periodically splitting a random subset of Gaussians regardless of their accumulated 2D gradient, thereby bypassing the densification gate that α -blending suppresses. Both operators are deliberately simple. We choose this minimal design because our goal is to verify that explicit exploration, on its own, is what 3DGS optimization is missing. A more sophisticated exploration strategy is certainly possible, but the burden of prove should first rest on the principle itself rather than on engineering. Experiments on Mip-NeRF 360 [12], Tanks & Temples [13], Deep Blending [14], OMMO [15], and DL3DV [16] confirm that each operator resolves its targeted form of blur, that the two operators are complementary, and that together they consistently improve rendering fidelity across all datasets with only marginal computational overhead.

Contributions of this paper can be summarized as:

1. We identify and formalize the **Blur Trap**, a fundamental optimization limitation of 3DGS that arises from gradient biases inherent to its physically derived back-propagation. We further categorize the Blur Trap into two subtypes, the Far-Side Blur Trap and the Near-

Side Blur Trap, each rooted in a different facet of the gradient bias.

2. We provide a rigorous theoretical analysis of the Blur Trap. In particular, we prove that the 3D position update derived from the 2D gradient is orthogonal to the viewing ray, which directly explains the absence of depth-directed optimization, and we characterize how α -blending systematically attenuates the densification signal for occluded primitives.
3. We propose two minimal exploration operators, **Random Seeding** and **Random Splitting**, targeting the Far-Side Blur Trap and Near-Side Blur Trap respectively. The minimal design is intentional, isolating the contribution of exploration itself from incidental engineering choices.
4. Extensive experiments on five different datasets demonstrate that each operator resolves its targeted form of blur, and that the two operators are complementary, jointly establishing exploration as a missing and essential ingredient in 3DGS optimization.

2. Related Work

2.1. Exploitation-Driven Gaussian Splatting Optimization

Adaptive Density Control in 3D Gaussian Splatting. Adaptive density control (ADC) is the cornerstone of 3D Gaussian Splatting (3DGS), dynamically adjusting primitive counts to balance geometric fidelity and computational efficiency. However, its strict reliance on view-space positional gradients introduces fundamental optimization bottlenecks, including signal dilution, directional conflicts, and premature stagnation. A primary line of research targets these gradient computation flaws. AbsGS [3] identifies gradient collisions and proposes homodirectional view-space gradients to mitigate directional conflicts. GDAGS [17] introduce a Gradient Coherence Ratio to measure directional consistency, enabling selective densification by distinguishing over-reconstructed from under-densified regions. Addressing scale bias, Pixel-GS [4] propose pixel-aware gradients to normalize magnitude variations caused by differing pixel footprints. Similarly, [18] replace heuristic gradient thresholds with pixel-error-driven criteria and correct opacity biases during cloning.

Modeling Distant Content in Unbounded Scenes. HoGS [7] addressed this by introducing homogeneous coordinates for position and scale, enabling unified near-far representation. Structural approaches have also proven effective: Scaffold-GS [19] established an anchor-based hierarchical representation with view-adaptive attribute decoding, naturally handling varying observation distances. Proxy-GS [20] further refined anchor placement by projecting anchors onto proxy mesh surfaces, preventing redundant growth behind occluders.

While these advances enhance densification robustness and extend 3DGS to unbounded domains via coordinate transfor-

mations or structural priors, they remain fundamentally bound to deterministic, gradient-driven exploitation. Stochastic alternatives that step outside this paradigm are comparatively rare, and we discuss them in the following section.

2.2. Exploration in Machine Learning

Exploration as a learning principle. The role of exploration is a recurring theme across non-convex optimization and reinforcement learning. When a purely greedy update rule operates on a complex landscape, it tends to commit prematurely to whatever direction the local signal points toward, missing solutions hidden behind barriers or in under-visited regions. Classical optimization addresses this with simulated annealing [21], while modern stochastic optimization adds gradient noise through Stochastic Gradient Langevin Dynamics (SGLD) [9] or perturbed gradient descent, which is shown to escape saddle points with near-dimension-free guarantees [10, 11]. Reinforcement learning takes the same idea further, treating exploration as a first-class algorithmic problem rather than an auxiliary mechanism [22].

Two families of exploration mechanisms. Exploration mechanisms in this broader literature divide into two complementary families. *Noise-based* methods inject stochasticity directly into the update rule, ranging from ϵ -greedy action selection to parameter-space perturbation that adds noise to network weights [23, 24]. *Coverage-driven* methods instead steer exploration by explicit estimates of novelty or uncertainty, as in count-based bonuses [25], prediction-error curiosity [26], and random network distillation [27]. The two families are not interchangeable: the former refines locally around the current state, while the latter allocates effort toward regions the model has never reached [28]. Which is more effective depends on the specific failure mode the learner faces.

Exploration in 3DGS optimization. Despite this rich literature, exploration remains a niche thread in 3DGS, where mainstream training follows a deterministic densify-and-prune cycle from a Structure-from-Motion initialization [1]. 3DGS-MCMC [8] is the closest predecessor of our perspective: it interprets Gaussians as samples from a distribution and converts position updates into SGLD steps via Langevin noise. A contemporaneous work, Opt3DGS [29], makes the exploration-exploitation framing explicit by alternating an adaptive SGLD phase with a curvature-aware exploitation phase. Both apply a single, uniform perturbation across all Gaussians without distinguishing between distinct failure modes. In contrast, we trace the difficulty back to two specific gradient biases identified in Section 3.3, and design two minimal operators each tied to one bias: **Random Seeding** acts as a coverage-driven mechanism that introduces new primitives in depth intervals unreachable from existing Gaussians, while **Random Splitting** acts as a noise-based mechanism that bypasses the densification gate suppressed by α -blending. Our work thereby treats exploration in 3DGS as a diagnosis-driven design problem rather than a uniform perturbation strategy.

3. Understanding the Blur Trap in 3D Gaussian Splatting

In Section 3, we present a systematic analysis of the optimization deficiencies in 3D Gaussian Splatting (3DGS) that lead to the Blur Trap. Specifically, Section 3.1 characterizes how the backpropagation in 3DGS induces an 2D gradient exploitation-dominant optimization regime. Section 3.2 then identifies the optimization bias of 3DGS. Finally, Section 3.3 analyses how these biases drives optimization into the Blur Trap.

3.1. Screen-Space Gradient Dominance in Optimization

3D Gaussian Splatting (3DGS) reconstructs 3D scenes through a differentiable rendering framework. The forward pass builds a rendering pipeline that projects 3D Gaussian primitives onto the image plane via a *physical camera model*, producing pixel colors through α -blending. Crucially, backpropagation derives gradients through this same differentiable pipeline, enabling direct optimization of Gaussian attributes against the reconstruction loss. Among these attributes, the 3D position of each Gaussian primitive is paramount: it determines where the primitive projects in screen space, governs how it blends with neighboring Gaussians, and ultimately defines the geometric structure of the reconstructed scene [30, 31].

The 3D position gradient of a Gaussian, $\mathbf{g}_{\text{all}}$, decomposes into three components:

$$\mathbf{g}_{\text{all}} = \mathbf{g}_{2d} + \mathbf{g}_{\text{cov}2d} + \mathbf{g}_{\text{sh}}, \quad (1)$$

where $\mathbf{g}_{2d}$ originates from the 2D screen-space projection, $\mathbf{g}_{\text{cov}2d}$ stems from the 2D covariance matrix governing the image-plane footprint, and $\mathbf{g}_{\text{sh}}$ derives from the spherical harmonic coefficients encoding view-dependent appearance. Conceptually, $\mathbf{g}_{2d}$ captures how the primitive should move to reduce reprojection error, while $\mathbf{g}_{\text{cov}2d}$ and $\mathbf{g}_{\text{sh}}$ convey complementary geometric and appearance signals.

These three components differ drastically in magnitude. As shown in Figure 2, $\mathbf{g}_{2d}$ is consistently orders of magnitude larger than $\mathbf{g}_{\text{cov}2d}$ and $\mathbf{g}_{\text{sh}}$. This disparity is structural: the 2D position gradient directly captures reprojection sensitivity to spatial displacements, whereas the covariance and SH gradients derive from less direct transformations. Consequently, $\mathbf{g}_{2d}$ dominates parameter updates, reducing the three-component optimization to greedily minimizing 2D reprojection error.

To verify this dominance, we ablate individual gradient components during Gaussian primitive updates. Models retaining $\mathbf{g}_{2d}$ maintain reconstruction performance comparable to standard 3DGS, confirming its sufficient role in driving optimization. Excluding $\mathbf{g}_{2d}$, however, leads to rapid quality degradation, while removing $\mathbf{g}_{\text{cov}2d}$ or $\mathbf{g}_{\text{sh}}$ individually produces negligible effect (Table 1). These results confirm that the optimization is effectively driven by screen-space positional signals alone, with the remaining components playing a marginal role. In sum-

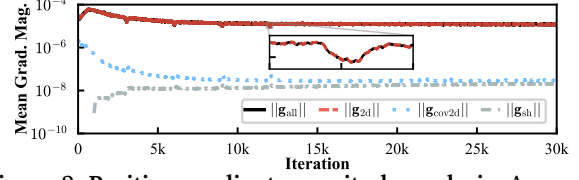


Figure 2: Position gradient magnitude analysis. Across the entire training process, $\|\mathbf{g}_{2d}\|$ remains almost identical to $\|\mathbf{g}_{\text{all}}\|$, while $\|\mathbf{g}_{\text{cov}2d}\|$ and $\|\mathbf{g}_{\text{sh}}\|$ are two to three orders of magnitude smaller, demonstrating that the 2D gradient component dominates 3D position gradient update.

Pos. Grad. Item			Mip-NeRF360 [12]			Tanks & Temples [13]			Deep Blending [14]		
Cov.	2D	SH	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
✓	✓	✓	27.52	0.816	0.215	23.73	0.853	0.169	29.80	0.907	0.238
✓	✓	✓	27.52	0.816	0.215	23.77	0.853	0.169	29.85	0.907	0.238
✓	✓	✓	24.27	0.661	0.386	21.59	0.729	0.333	24.92	0.825	0.376
✓	✓	✓	27.53	0.816	0.215	23.73	0.853	0.169	29.78	0.907	0.239
✓	✓	✓	24.18	0.658	0.388	21.54	0.724	0.336	24.88	0.826	0.376
✓	✓	✓	27.54	0.816	0.215	23.84	0.853	0.170	29.80	0.907	0.238
		✓	25.23	0.706	0.340	22.45	0.771	0.285	28.12	0.879	0.301

Table 1: Ablation study on 3D position gradient items. We evaluate the impact of different gradient items on rendering quality. 2D position gradient branch is a key branch for 3DGS optimization. Pos. Grad. Item denote Position Gradient Item.

mary, although the 3D position update formally combines three gradient contributions, **their severe magnitude disparity degenerates the optimization into a greedy, 2D screen-space gradient exploitation process.** The optimizer minimizes reprojection errors through $\mathbf{g}_{2d}$ while effectively ignoring the complementary geometric and appearance signals from $\mathbf{g}_{\text{cov}2d}$ and $\mathbf{g}_{\text{sh}}$. As a result, Gaussian primitives rapidly converge to configurations that locally minimize pixel error. This is a classic exploitation behavior that prioritizes immediate screen-space fidelity at the expense of global structural integrity. **This analytical insight naturally leads to the central question: does such a greedy 2D optimization strategy introduce systematic reconstruction failures in 3DGS?**

3.2. Depth Gradient Deficiency and Blending-Induced Den-sification Failure

The 2D position gradient in 3DGS fulfills a dual role: it drives 3D positional updates and serves as the core criterion for adaptive density control (ADC) [1]. Consequently, this single optimization branch exerts disproportionate control over the entire reconstruction pipeline. We systematically analyze the inherent pitfalls of greedily relying on the 2D gradient for both position optimization and densification. Our analysis establish a mechanistic basis for understanding why standard 3DGS frequently stagnates in the *Blur Trap*. We demonstrate that this exploitation-heavy strategy inherently lacks depth-directed supervision. When coupled with the α -blending pipeline, the periodically accumulated 2D gradients used for densification are systematically diluted. *The detailed derivation is provided in the Appendix A and B.*

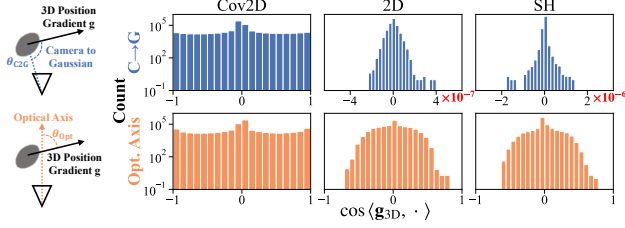


Figure 3: Gradient Direction Analysis: Orthogonality and Depth-Signal Deficiency. Histograms of cosine similarity between the three 3D position gradient branches and the reference directions. **Top:** Relative to the viewing ray. The 2D position and SH branches are orthogonal to the viewing ray, confirming that screen-space reprojection errors cannot produce depth-directed updates. **Bottom:** Relative to the optical axis. Limited to field-of-view, optimization directions are mostly distributed between 60° and 120° relative to the optical axis, indicating a severe lack of depth-directed signals along the optical axis. The Cov2D branch exhibits more diverse directional distribution, including depth-aligned components. However, its gradient magnitude is orders of magnitude smaller than that of the 2D position branch (Section 3.1), rendering it ineffective for depth optimization. This directional bias, combined with the magnitude imbalance, systematically denies 3DGS effective depth-directed gradient signals.

3.2.1. Depth Gradient Deficiency

We prove that the 3D positional update direction of any Gaussian primitive remains strictly orthogonal to the viewing ray from the camera center to the primitive. This geometric constraint is formalized as:

$$\frac{\partial L^{2D}}{\partial \mathbf{P}_{3D}} (\mathbf{P}_{3D} - \mathbf{P}_{\text{cam}}) = 0, \quad (2)$$

where $\frac{\partial L^{2D}}{\partial \mathbf{P}_{3D}}$ denotes the 3D position gradient component derived exclusively from the 2D position gradient branch, $\mathbf{P}_{3D}$ represents the Gaussian’s 3D position, and $\mathbf{P}_{\text{cam}}$ indicates the camera optical center.

Translating a Gaussian along the viewing ray doesn’t induce any change in its projected 2D screen position. Therefore, to minimize the loss from projection, the optimizer consistently directs 3D positional updates orthogonal to this ray. Given typically narrow camera fields of view and centrally framed subjects, viewing rays align closely with the optical axis. Consequently, the orthogonal update directions lie nearly parallel to the image plane. This geometric configuration inherently starves the optimization process of depth-directed signals.

We empirically validate this orthogonality and the resulting depth-gradient deficiency. As illustrated in Figure 3, we extract the 3D positional update vectors derived from three distinct gradient branches. We then compute their cosine similarities relative to both the viewing ray and the camera optical axis. The results confirm that updates driven by the 2D position

Branch	Magnitude	Direction
$\mathbf{g}_{2d}$	$\approx \ \mathbf{g}_{\text{all}}\ $	Mathematically orthogonal to the viewing ray
$\mathbf{g}_{\text{all}}$	$\ll \ \mathbf{g}_{\text{all}}\ $	Experientially orthogonal to the viewing ray
$\mathbf{g}_{\text{cov2d}}$	$\ll \ \mathbf{g}_{\text{all}}\ $	More random direction

Table 2: Magnitude and direction of the three gradient branches. The 2D position gradient $\mathbf{g}_{2d}$ dominates the total position gradient in magnitude and is strictly orthogonal to the viewing ray, while the spherical harmonic and 2D covariance gradients contribute negligibly to the 3D position update.

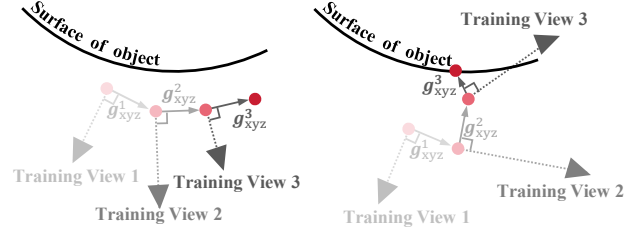


Figure 4: Geometric Regularization via Multi-View Consistency under Gradient Orthogonality. 3DGS optimizes Gaussian positions along directions strictly orthogonal to the viewing ray. **Left:** Rich multi-view observations provide intersecting constraints that implicitly regularize primitives onto the true object surface. **Right:** Under insufficient view diversity, this geometric regularization fails. 3DGS compensates by proliferating redundant primitives to fit limited view directions, leading to severe overfitting and structural artifacts. Note that this is a simplified 2D illustration; in full 3D space, the absence of supervisory views from any single direction leads to incomplete geometric optimization.

gradient remain orthogonal to the Camera-to-Gaussian ray. Moreover, their cosine values against the optical axis cluster tightly around zero. As summarized in Table 2, the angular and magnitude characteristics confirm that the 3D position update is dominated by the 2D position gradient branch and remains orthogonal to the viewing ray. Consequently, optimization signals along the depth axis are substantially weaker than those within the image plane, and effective 3DGS reconstruction fundamentally depends on diverse camera trajectories. As shown in Figure 4, multi-view observations supply intersecting constraints that collectively resolve depth ambiguities.

In principle, capturing an object from all surrounding view-points provides sufficient directional supervision for full geometric optimization. In practice, however, acquiring multi-view coverage of every scene element is often prohibitively expensive (e.g., distant mountains in background) or physically impossible (e.g., clouds in the sky). In these cases, although the training set includes multiple views, the corresponding camera-to-object ray directions cluster within a narrow angular range. This limited viewing configuration deprives distant Gaussian primitives of depth-directed optimization signals, leading to reconstruction degradation in background regions [7].

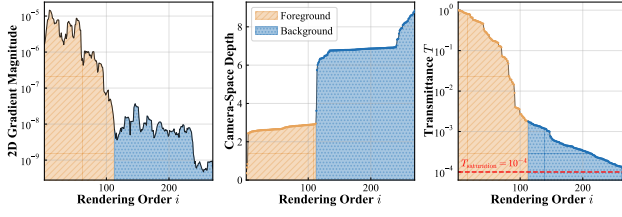


Figure 5: Blending-Induced Gradient Attenuation and Color-Structure Inconsistency. The 2D position gradient magnitude, camera-space depth, and transmittance T of primitives compositing a single pixel, plotted against the rendering order i . **Left:** Foreground primitives ($i \leq 112$) maintain strong gradient signals, while background primitives ($i > 112$) exhibit drastically attenuated magnitudes due to cumulative transmittance decay. **Middle:** Foreground and background primitives occupy distinct depth intervals. Depth distribution reveals that although they jointly contribute to the same pixel color, they represent spatially disjoint geometric structures. **Right:** Although foreground primitives significantly reduce accumulated transmittance T , α -blending continues compositing background Gaussians until $T < T_{\text{saturation}}$ (0.0001). This forces background primitives to density and optimize position with severely attenuated gradients.

In summary, 3DGS optimization functions as an inherently local, position-driven process. Although the framework employs adaptive densification, it fundamentally lacks optimization momentum along the viewing ray. This absence of depth-directed exploration systematically constrains the effectiveness of the densification pipeline.

3.2.2. Blending-Induced Densification Failure

The 2D position gradient $\frac{\partial L}{\partial \mathbf{p}_{2D}}$ also acts as the screening metric for adaptive densification. Prior to each densification step, primitives exhibiting substantial gradient magnitudes on the image plane are flagged as spatially active and selected for densification. However, the α -blending pipeline introduces a systematic gradient bias. **During backpropagation, α -blending suppresses densification by delivering severely weakened 2D position signals to occluded regions.**

$$C_p = \sum_{i=1}^N c_i \alpha_i T_i, \quad T_i = \prod_{j=1}^{i-1} (1 - \alpha_j) \quad (3)$$

As formalized in Equation 3, where i denote the rendering order of a Gaussian within the α -blending sequence. The primitives rendered later in the depth-sorted sequence contribute minimally to the final pixel color due to accumulated foreground opacity [32]. Consequently, the gradient backpropagated to these later-ranked Gaussians is consistently attenuated compared to that received by foreground primitives [33]. The transmittance T_i effectively weights the contribution of primitive i to both the rendered output and the

loss computation. **Since T_i decays rapidly as i increases, the 2D position optimization signal for occluded Gaussians is progressively suppressed.**

Rendering a single pixel typically requires compositing hundreds of Gaussian primitives via α -blending [34], governed by a predefined opacity termination criterion. However, these primitives that co-render the same pixel do not necessarily correspond to a single coherent 3D surface [20], which means that the rendered pixels are inconsistent with the actual 3D structure [35, 36]. Consequently, occluded background Gaussians share the same pixel as foreground primitives. Due to the multiplicative transmittance term in the α -blending pipeline, gradient signals backpropagated to these occluded primitives are severely attenuated relative to foreground updates.

We empirically validate this disparity by profiling the primitives contributing to a representative central pixel. Specifically, we record their 2D position gradient magnitudes and camera-space depths. As illustrated in Figure 5, primitives rendered earlier in the front-to-back sequence exhibit substantially larger gradient magnitudes than later-ranked ones. The corresponding depth visualization further reveals that these contributing Gaussians span three distinct depth intervals. Crucially, adaptive densification in 3DGS relies on periodically averaged 2D position gradients. For occluded primitives, this periodic statistic is systematically diluted by consistently weak signals across multiple views. As a result, the accumulated gradient fails to reach the splitting threshold, directly causing densification failure in background regions.

3.3. Formation of the Blur Trap

Collectively, screen-space gradient dominance (Section 3.1), depth gradient deficiency, and blending-induced densification failure (Section 3.2) establish an **exploitation-only optimization dynamic**. This dynamic fundamentally shapes the optimization landscape of 3DGS, consistently driving the system into persistent local optima.

We identify this systematic optimization stagnation as a fundamental limitation intrinsic to 3DGS, which we formally define as the **Blur Trap**. As illustrated in Figure 6, pronounced blurring artifacts persist in distant regions (e.g., mountains) and occluded areas (e.g., beneath chairs), despite abundant multi-view supervision in the training views. Based on their distinct formation mechanisms, we categorize the Blur Trap into two variants: the *Far-Side Blur Trap* and the *Near-Side Blur Trap*.

3.3.1. Optimization Dynamics and Trap Formation.

Far-Side Blur Trap. The 3D position optimization in 3DGS inherently lacks depth-directed supervision in camera space. Consequently, Gaussian primitives fail to converge to their correct spatial locations along the viewing ray. This chronic absence of depth optimization signals directly manifests as the Far-Side Blur Trap.

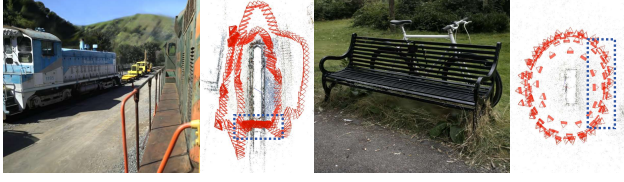


Figure 6: Blur Trap under Sufficient Supervision. Far-Side Blur Trap (Left): Distant regions (mountains) exhibit persistent structural smoothing even with abundant training views covering these areas. **Near-Side Blur Trap (Right):** Occluded near-field areas (beneath the bench) lack structural detail despite adequate view coverage. The dashed boxes in top-view camera pose distributions denote dense supervision views for the blurred areas.

Near-Side Blur Trap. Although foreground primitives typically occupy more pixels and benefit from broader multi-view coverage, they frequently act as background elements in specific views. Under occlusion, the α -blending pipeline systematically attenuates their 2D position gradients. Consequently, the periodically averaged gradient magnitude consistently falls below the densification threshold. This splitting stagnation traps local geometry in suboptimal configurations, ultimately forming the Near-Side Blur Trap.

3.3.2. Motivation for Explicit Exploration.

The Blur Trap is not a transient optimization phase but an inherent consequence of the backpropagation pipeline. Deterministic gradient descent cannot escape this basin because the required depth-probing signals *are mathematically suppressed by the rendering formulation itself*. Escaping this stagnation necessitates explicit perturbations.

4. Methodology

In Section 4.1, we first establish targeted exploration principles based on the formation mechanisms of the Blur Trap. Then, in Section 4.2 and Section 4.3, we introduce two minimal exploration operators designed to escape the Far-Side and Near-Side Blur Trap, respectively.

4.1. Exploration Principles

Building on the analysis in Section 3, we identify the root cause of the Blur Trap as the physics-based formulation of the differentiable rendering pipeline. Because both parameter updates and adaptive densification rely exclusively on backpropagated gradients, the optimization process inherits a systematic bias. Specifically, the overwhelming dominance of the 2D position gradient reduces 3D position optimization to a greedy, screen-space exploitation process. This gradient-dependent dynamic inherently neglects depth-ambiguous and

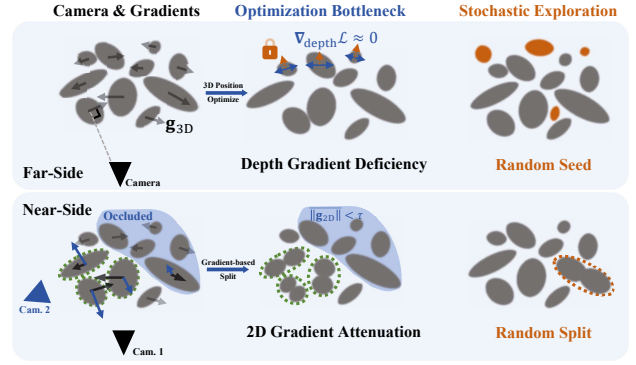


Figure 7: Stochastic Exploration for Escaping the Blur Trap. Far-Side (Top): Physically-derived 3D position gradients $\mathbf{g}_{3D}$ are strictly orthogonal to the viewing direction, yielding zero update signals along the depth axis ($\nabla_{\text{depth}} \mathcal{L} \approx 0$) in distant zone. *Random Seed* injects **sparse Gaussian primitives** directly to probe unexplored depths. **Near-Side (Bottom):** In **occluded regions**, α -blending attenuation suppresses gradient magnitudes below the densification threshold ($\|\mathbf{g}_{2D}\| < \tau$). Consequently, standard **gradient-based splitting** fails to allocate representational capacity to structurally important primitives. *Random Split* relaxes this deterministic mask via probabilistic splitting, **random splitting Gaussians** to restore detail.

occluded regions, ultimately trapping the reconstruction in suboptimal configurations.

To break this, our core design principle is to **bypass purely gradient-driven exploitation by injecting explicit exploration**. Rather than waiting for suppressed signals to recover, we proactively introduce perturbations that operate independently of the 2D gradient magnitude. Specifically, for the Far-Side Blur Trap, exploration must establish update pathways that propagate primitives into distant regions along the depth axis. For the Near-Side Blur Trap, it must proactively trigger densification for primitives whose 2D gradients consistently fall below the standard split threshold τ_{split} .

To rigorously validate the intrinsic efficacy of exploration, we deliberately adopt the simplest possible operators, prioritizing conceptual simplicity over engineering refinements. Accordingly, we introduce two minimal exploration operators: *Random Seeding* and *Random Splitting*, detailed in Section 4.2 and Section 4.3, which directly target the *Far-Side* and *Near-Side Blur Trap*, respectively.

4.2. Random Seeding for Far-Side Exploration

Guided by the principle of establishing depth-directed pathways, the *Random Seeding* operator directly stochastically samples a sparse set of seed Gaussians in 3D space and injects them into the existing primitive collection, as illustrated in the top row of Figure 7. This intervention does not violate the viewing-gradient orthogonality inherent to 3DGS. By distribut-

Method	Mip-NeRF360 [12]			Tanks & Temples [13]			Deep Blending [14]			OMMO [15]			DL3DV Benchmark [16]		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
3DGS*	27.52	0.816	0.215	23.73	0.853	0.169	29.80	0.907	0.238	30.49	0.920	0.142	27.16	0.860	0.167
HoGS* (50K Iteration)	27.55	0.818	0.201	24.23	0.859	0.160	29.21	0.898	0.244	30.66	0.919	0.135	28.16	0.878	0.140
Seed Explore	27.68	0.819	0.214	24.30	0.861	0.163	29.78	0.906	0.237	30.85	0.924	0.131	27.74	0.869	0.155
Split Explore	27.95	0.829	0.192	24.30	0.868	0.139	30.01	0.908	0.248	31.29	0.928	0.121	28.47	0.889	0.125
Seed & Split Explore	27.96	0.829	0.195	24.37	0.870	0.139	29.98	0.907	0.249	31.27	0.930	0.119	28.43	0.891	0.124

Table 3: **Reconstruction Quality Across Different Datasets.** *Seed Exploration* consistently improves unbounded scenes (e.g., T&T), while *Split Exploration* excels in occluded regions (e.g., OMMO). Their combination achieves optimal performance in most scenarios, validating simultaneous resolution of both Blur Trap variants. (*All experiments rerun based on the same data.)

Method	Mip-360	T & T	DB	OMMO	DL3DV
3DGS	2.72	1.57	2.48	1.78	1.14
HoGS (50K)	4.40	2.16	2.43	2.01	1.61
Seed Exp.	2.62	1.51	2.33	1.68	1.11
Split Exp.	2.58	2.13	0.86	1.77	2.13
Seed & Split Exp.	2.53	2.11	0.79	1.77	2.11

Table 4: **Gaussian primitive allocation (in millions) across different models.** *Seed Exploration* (*Seed Exp.*) marginally reduces the primitive count compared to 3DGS. On the Deep Blending (DB) dataset, *Split Exploration* (*Split Exp.*) drastically decreases Gaussian count. Conversely, for complex scenes like Tanks & Temples (T&T) and DL3DV, the strategy adaptively increases primitive allocation to resolve fine geometric details, resulting in a substantial fidelity gain (Table 3).

ing seeds uniformly across the 3D space, the operator explicitly probes depth intervals that are inaccessible to gradient-based updates, thereby compensating for the lack of depth-directed optimization signals in the differentiable framework.

The sparse injection schedule ensures minimal disruption to well-reconstructed regions, preserving the stability of the ongoing optimization. Seeds placed in invalid or empty areas fail to reduce the photometric loss and are automatically removed by the native opacity pruning mechanism. Conversely, seeds landing in geometrically plausible regions are guided by the standard 2D reprojection loss. They refine their positions within the plane orthogonal to the viewing ray, eventually triggering adaptive densification. This establishes a complementary optimization loop: **Random Seeding supplies global depth exploration, while deterministic gradients drive local planar exploitation.**

We characterize this operator as a global exploration mechanism. Its primary function is to overcome the fundamental limitation of 3DGS: the inability of deterministic, locally-focused position updates to explore uncharted depth dimensions.

4.3. Random Splitting for Near-Side Exploration

As illustrated in the bottom row of Figure 7, the *Random Splitting* operator bypasses the gradient-dependent densification criterion by proactively targeting large-scale Gaussians across the entire scene for exploratory splitting. Specifically, we sample primitives based on their mean scale and split only a small subset (N_{split}) per iteration to preserve training stability. This

strategy allows occluded regions to circumvent the suppression of 2D gradients and actively participate in the densification process. Once these regions are populated with finer primitives, they rapidly converge to optimal configurations under unoccluded training views.

Random Splitting directly addresses the fundamental limitation of 3DGS: in occluded areas, the contribution to foreground color is attenuated by α -blending, which dilutes the 2D position signals. Consequently, these primitives fail to reach the statistical splitting threshold, preventing effective densification. ***Random Splitting* restores this capability by decoupling structural refinement from gradient magnitude.**

5. Experiments

5.1. Experimental Settings

5.1.1. Implementation Details

We integrate these operators into standard 3DGS adhering to the principle of minimal intervention.

For *Random Seeding*, the operator uniformly samples N_{seed} candidate positions at each densification iteration within the minimum bounding box of all existing Gaussians ($N_{\text{seed}} = 20$ by default). For *Random Splitting*, the operator randomly selects N_{split} large-scale Gaussians to split per iteration ($N_{\text{split}} = 20$ by default). Although N_{seed} and N_{split} are negligible compared to the total primitive count (often in the millions), this sparse exploration proves highly effective in escaping the Blur Trap.

5.1.2. Dataset and Metrics

We comprehensively evaluate our method across multiple 3D reconstruction benchmarks, including Mip-NeRF 360 [12], Tanks & Temples [13], Deep Blending [14], the large-scale outdoor dataset OMMO [15], and the DL3DV Benchmark [16]. We randomly select a subset of scenes from the DL3DV Benchmark for detailed evaluation with $\frac{1}{4}$ resolution images¹. It is important to note that the Blur Trap is not unique to these selected scenes; rather, it is a widespread phenomenon observed across unselected scenarios in all tested datasets. Reconstruction performance is quantified using PSNR, SSIM, and LPIPS.

¹DL3DV Scene IDs: 1, 101, 104, 109, 11, 21, 24, 26, 34, 42, 69, 7, 8, 82, 90, 97

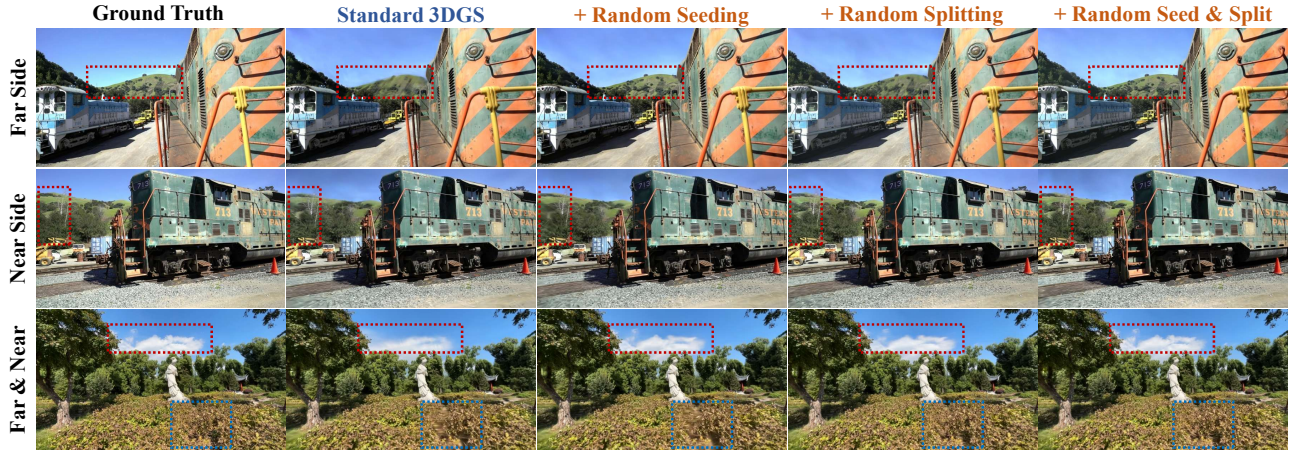


Figure 8: Complementarity of Random Seeding and Random Splitting. **Top:** Standard 3DGS produces persistent blur in far-side regions, while *Random Seeding* effectively recovers these details by enabling depth-directed exploration. **Middle:** 3DGS suffers from blur in occluded near-side regions, while *Random Splitting* successfully restores structural sharpness by decoupling densification from gradient magnitude. **Bottom or fourth column:** *Random Seeding* and *Random Splitting* complement each other to escape both Blur Trap variants, recovering clear details in far-side backgrounds and occluded near-side regions.

Additionally, the number of Gaussian primitives is monitored to assess computational efficiency.

5.2. Experiments Results

Quantitative results across multiple datasets confirm the efficacy of our minimal exploration operators. As shown in Table 3, both *Seed Exploration* and *Split Exploration* consistently improve reconstruction fidelity, with their combination achieving optimal performance across most benchmarks. This validates that the Far-Side and Near-Side Blur Traps can be simultaneously resolved through exploration. The visual comparison is shown in Figure 12 and Figure 13.

The Deep Blending (DB) dataset offers a critical insight. While *Split Exploration* achieves the highest PSNR and SSIM, its LPIPS score is slightly suboptimal. This trade-off is directly explained by Table 4, which reveals a 65% reduction in Gaussian primitives compared to the 3DGS baseline, while maintaining superior geometric fidelity. This confirms that the operator prioritizes structural efficiency over redundant primitives, effectively resolving the Near-Side Blur Trap without introducing unnecessary complexity.

Crucially, *Seed Exploration* requires only minimal modifications to the standard 3DGS pipeline. It slightly reduces the total primitive count while consistently improving PSNR, demonstrating that simple stochastic exploration is highly effective. Similarly, *Split Exploration* reduces primitives in simpler regions (e.g., DB) while adaptively increasing them in geometrically complex scenes, a favorable trade-off that allocates computation where it most improves fidelity. This adaptive allocation reduces redundancy in simple areas while refining complex structures, enabling our operators to overcome

the Near-Side Blur Trap. The visual comparison in Figure 8 confirms complementarity: *Random Seeding* recovers far-side background details that 3DGS leaves blurry, while *Random Splitting* restores sharpness in occluded near-side regions. Their joint application (fourth column) simultaneously resolves both types of degradation, demonstrating that the two operators target distinct failure modes of the 3DGS optimization.

Together, these results confirm that minimal stochastic exploration is sufficient to escape the Blur Trap by disrupting its gradient-driven optimization dynamics. Our operators consistently improve fidelity while maintaining or reducing model complexity, establishing that **the Blur Trap arises from a fundamental optimization bias in 3DGS: heavy exploitation without explicit exploration.**

5.3. 4D Gaussian Splatting Blur Trap

4D Gaussian Splatting Due to the shared optimization pipeline, the Blur Trap inherently propagates from 3DGS to 4D Gaussian Splatting (4DGS) [37]. We validate this limitation on a representative 4DGS framework with Neu3D [38] dataset. As illustrated in Figure 9, despite extensive multi-view coverage, regions classically susceptible to occlusion remain severely blurred in the baseline reconstruction, such as distant buildings visible through windows, text on indoor bottles, and steak textures dynamically obscured by flames. These areas suffer from suppressed 2D positional gradients, which systematically inhibits adaptive densification.

We integrating *Random Splitting* with a highly sparse splitting ($N_{\text{split}} = 5$), and it successfully bypasses the Blur Trap to trigger effective densification. Occluded primitives actively participate in structural refinement and detail recovery. As



Figure 9: Random Exploration in 4D Gaussian Splatting. **Left:** 4DGS inherits the Blur Trap, failing to densify occluded regions. Background structures (e.g., buildings behind windows) receive severely attenuated optimization signals that systematically suppress densification. Similarly, dynamic occluders such as flames hinder the refinement of food textures. **Middle and Right:** By integrating a highly sparse *Random Splitting* ($N_{\text{split}} = 5$), these stagnated primitives bypass gradient-dependent thresholds and undergo structural refinement. This minimal intervention yields substantial detail recovery across both static and dynamic occlusions, confirming that explicit exploration transfers to 4DGS pipelines.

Method	PSNR $\uparrow$	SSIM $\uparrow$	MS-SSIM $\uparrow$	D-SSIM $\downarrow$	LPIPS-vgg $\downarrow$	LPIPS-alex $\downarrow$
4DGS	30.575	0.9314	0.9659	0.0171	0.1507	0.0602
Split Exp.	30.820	0.9369	0.9689	0.0156	0.1380	0.0488
Split & Seed Exp.	31.085	0.9379	0.9702	0.0149	0.1377	0.0483

Table 5: 4DGS on the Neu3D Dataset. *Random Splitting* delivers the primary fidelity gains. Occlusion-induced near side blur remains the optimization bottleneck in Neu3D [38]. These scenes are captured within a relatively bounded depth range, which naturally limits the effectiveness of *Random Seeding*.

quantified in Table 5, these gains are most pronounced in perceptual fidelity (LPIPS). The fact that such a minimal intervention resolves severe blur confirms that standard splitting mechanisms, which strictly rely on 2D positional gradients, are fundamentally bottlenecked by signal attenuation. Explicit exploration, even at extremely low intensity, provides a more robust pathway for occluded-region refinement.

5.4. Ablation Studies

Random Seeding To validate the critical role of depth-directed signaling, we conduct an experiment using heuristic depth supervision. We generate pseudo-depth maps for all input views using Depth Anything V2 [39] and integrate a depth regularization term ($\mathcal{L}_{\text{Depth}}$) into the standard 3DGS pipeline.

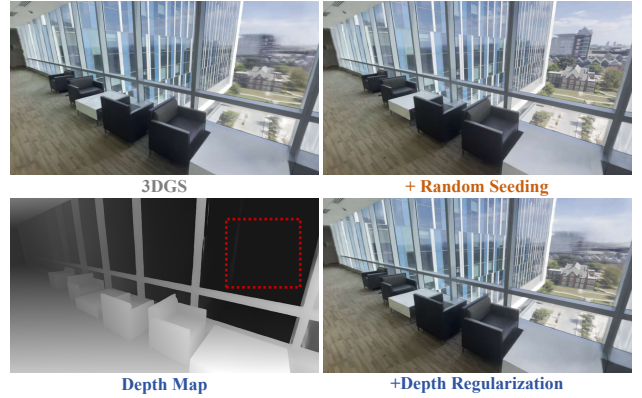


Figure 10: Depth Supervision versus Random Seeding for Far-Side Recovery. **Top-Left:** Standard 3DGS suffers from the Far-Side Blur Trap due to absent depth-directed optimization signals. **Bottom:** Incorporating pseudo-depth supervision partially alleviates distant blur, but its inherent inaccuracy restricts the effective range, leaving farther regions trapped. **Top-Right:** In contrast, *Random Seeding* explicitly probes uncharted depth intervals, enabling robust escape from the Far-Side Blur Trap across varying distances.

$\mathcal{L}_D$	Method	Tanks & Temples			OMMO			DL3DV Benchmark		
		PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
✗	3DGS	23.73	0.853	0.169	30.49	0.920	0.142	27.16	0.860	0.167
	Seed Exp.	24.30	0.861	0.163	30.85	0.924	0.131	27.74	0.869	0.155
	Seed & Split Exp.	24.37	0.870	0.139	31.27	0.930	0.119	28.43	0.891	0.124
✓	3DGS	24.06	0.856	0.166	30.59	0.923	0.136	27.33	0.864	0.162
	Seed Exp.	24.19	0.859	0.162	30.81	0.925	0.132	27.76	0.865	0.155
	Seed & Split Exp.	24.38	0.869	0.140	31.06	0.928	0.123	28.39	0.888	0.127

Table 6: Comparative Analysis: Deterministic Depth Supervision and Stochastic Exploration. $\mathcal{L}_D$ ($\mathcal{L}_{\text{Depth}}$) integrates pseudo depth maps as a regularization term into the original loss function. While incorporating $\mathcal{L}_D$ into 3DGS yields consistent fidelity gains, it actively suppresses stochastic exploration dynamics. Notably, sparse *Random Seeding* alone delivers superior fidelity improvements compared to deterministic depth regularization, confirming that explicit exploration is more robust than depth priors for resolving depth ambiguities.

Quantitative results across multiple unbounded scenes are summarized in Table 6.

Incorporating $\mathcal{L}_{\text{Depth}}$ into the 3DGS consistently improves reconstruction fidelity, empirically confirming that the absence of depth-directed supervision is a primary driver of the Far-Side Blur Trap. However, pseudo-depth estimation is inherently imprecise. Its fidelity gains remain inferior to those achieved by sparse Random Seeding operator. More critically, integrating $\mathcal{L}_{\text{Depth}}$ into stochastic exploration yields diminishing returns. The deterministic exploitation of these inaccurate depth signals actively suppresses stochastic exploration dynamics, as the optimizer becomes constrained by unreliable priors. As illustrated in Figure 10, while integrating pseudo-depth supervision partially mitigates the Far-Side Blur, the inherent inaccuracy of pseudo-depth cues restricts its effective range.

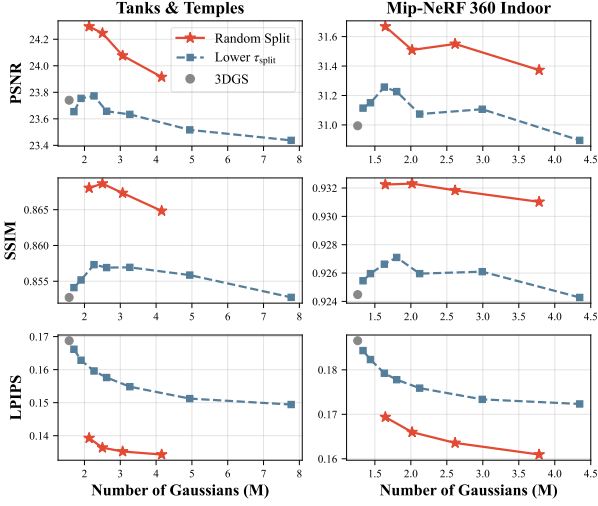


Figure 11: Comparison of reconstruction fidelity against the number of Gaussian primitives. The dashed line represents standard 3DGS with lowered splitting threshold τ_{split} , while solid line denote *Random Splitting* at varying N_{split} . Lower τ_{split} forces excessive splitting, leading to a sharp increase in primitive count. *Random Splitting* proactively targets large-scale Gaussians regardless of gradient suppression. Sparse random splits outperforms exploitation-only splitting.

Consequently, distant regions remain vulnerable to persistent blurring as the optimizer becomes constrained by unreliable geometric priors. In contrast, *Random Seeding* bypasses this limitation by explicitly probing uncharted depth intervals.

These demonstrate that minimal random exploration provides a more robust and effective pathway to escape the Blur Trap than forcing gradient descent along heuristic depth regularization. *An alternative depth exploration strategy is detailed in the Appendix C, further underscoring the critical role of explicit exploration.*

Random Splitting To verify that the fidelity gains from *Random Splitting* do not merely stem from an increased Gaussian primitive count, we directly compare its efficiency against standard gradient-driven densification by evaluating the trade-off between primitive allocation and reconstruction fidelity. Specifically, we test *Random Splitting* across varying intensities ($N_{\text{split}} \in \{20, 50, 100, 200\}$). And we lower the 2D position gradient threshold τ_{split} on 3DGS. As shown in Figure 11, simply reducing τ_{split} fails to yield meaningful fidelity improvements and instead introduces excessive redundant primitives. In contrast, **even a sparse application of *Random Splitting* efficiently enhances reconstruction quality** while maintaining a significantly leaner primitive allocation. This confirms that the performance gain originates from targeted structural exploration rather than brute-force densification.

6. Conclusion

We identify the *Blur Trap* as a fundamental optimization limitation in 3DGS, rooted in viewing-ray gradient orthogonality and blending-induced gradient attenuation that systematically suppress depth-directed signals. To break this exploitation-only deadlock, we introduce two minimal exploration operators: *Random Seeding* and *Random Splitting*. By bypassing gradient-dependent criteria, these operators proactively refine distant and occluded regions. Cross-benchmark evaluations confirm that their combination consistently escapes both Blur Trap, delivering state-of-the-art fidelity with negligible overhead and adaptive primitive allocation. This demonstrates that reconstruction bottlenecks stem from optimization dynamics, not model capacity. Our work establishes explicit exploration as a necessary component for differentiable rendering, advocating a shift toward balanced exploration-exploitation dynamics.



Figure 12: Visual comparison of escaping Far-Side Blur Trap for different models.

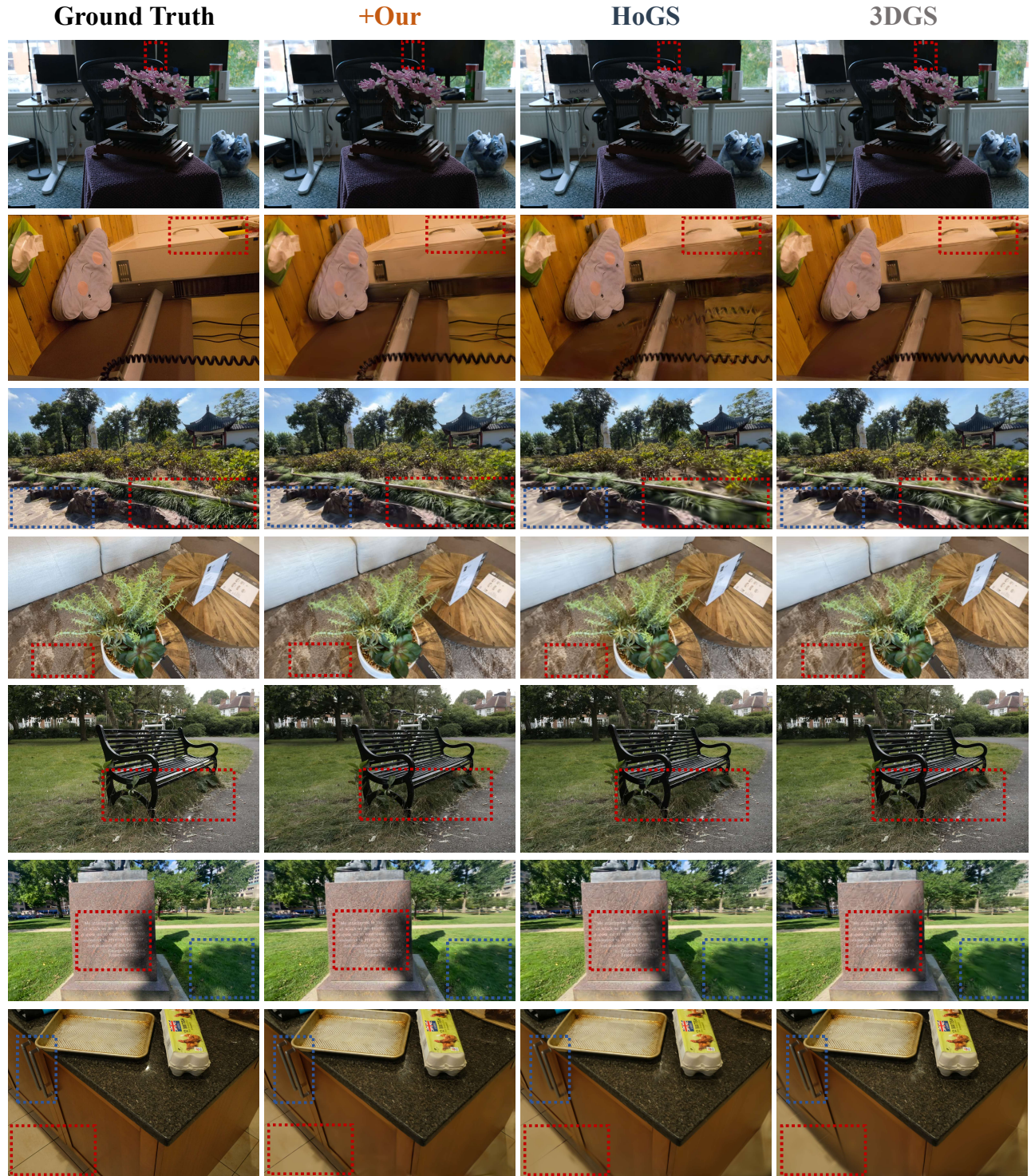


Figure 13: Visual comparison of escaping Near-Side Blur Trap for different models.

References

- [1] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. [1](#), [3](#), [4](#)
- [2] 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. [1](#)
- [3] Zongxin Ye, Wenyu Li, Sidun Liu, Peng Qiao, and Yong Dou. Absgs: Recovering fine details in 3d gaussian splatting. In *Proceedings of the 32nd ACM international conference on multimedia*, pages 1053–1061, 2024. [2](#), [3](#)
- [4] Zheng Zhang, Wenbo Hu, Yixing Lao, Tong He, and Hengshuang Zhao. Pixel-gs: Density control with pixel-aware gradient for 3d gaussian splatting. In *European Conference on Computer Vision*, pages 326–342. Springer, 2024. [2](#), [3](#)
- [5] Saswat Subhajyoti Mallick, Rahul Goel, Bernhard Kerbl, Markus Steinberger, Francisco Vicente Carrasco, and Fernando De La Torre. Taming 3dgs: High-quality radiance fields with limited resources. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024. [2](#)
- [6] Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of gaussians. In *European conference on computer vision*, pages 165–181. Springer, 2024. [2](#)
- [7] Xinpeng Liu, Zeyi Huang, Fumio Okura, and Yasuyuki Matsushita. Hogs: Unified near and far object reconstruction via homogeneous gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26714–26722, 2025. [2](#), [3](#), [5](#)
- [8] Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Yang-Che Tseng, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 3d gaussian splatting as markov chain monte carlo. *Advances in Neural Information Processing Systems*, 37:80965–80986, 2024. [2](#), [3](#), [21](#)
- [9] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011. [2](#), [3](#)
- [10] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. In *International conference on machine learning*, pages 1724–1732. PMLR, 2017. [2](#), [3](#)
- [11] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on learning theory*, pages 797–842. PMLR, 2015. [2](#), [3](#)
- [12] 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. [2](#), [4](#), [8](#)
- [13] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017. [2](#), [4](#), [8](#)
- [14] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)*, 37(6):1–15, 2018. [2](#), [4](#), [8](#)
- [15] Chongshan Lu, Fukun Yin, Xin Chen, Wen Liu, Tao Chen, Gang Yu, and Jiayuan Fan. A large-scale outdoor multi-modal dataset and benchmark for novel view synthesis and implicit scene reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7557–7567, 2023. [2](#), [8](#)
- [16] 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. [2](#), [8](#)
- [17] Zheng Zhou, Yu-Jie Xiong, Jia-Chen Zhang, Chun-Ming Xia, Xihe Qiu, and Hongjian Zhan. Gradient-direction-aware density control for 3d gaussian splatting. In *The Fourteenth International Conference on Learning Representations*, 2026. URL <https://openreview.net/forum?id=6qDxK4Gz7F>. [3](#)
- [18] Samuel Rota Bulò, Lorenzo Porzi, and Peter Kotschieder. Revising densification in gaussian splatting. In *European Conference on Computer Vision*, pages 347–362. Springer, 2024. [3](#)
- [19] 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. [3](#)
- [20] Yuanyuan Gao, Yuning Gong, Yifei Liu, Jingfeng Li, Dan Xu, Yanci Zhang, Dingwen Zhang, Xiao Sun, and Zhihang Zhong. Proxy-gs: Unified occlusion priors for training and inference in structured 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer*

- Vision and Pattern Recognition*, pages 7330–7339, 2026. 3, 6
- [21] Scott Kirkpatrick, C Daniel Gelatt Jr, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220 (4598):671–680, 1983. 3
- [22] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998. 3
- [23] Matthias Plappert, Rein Houthoofd, Prafulla Dhariwal, Szymon Sidor, Richard Y Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter space noise for exploration. *arXiv preprint arXiv:1706.01905*, 2017. 3
- [24] Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Matteo Hessel, Ian Osband, Alex Graves, Volodymyr Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, Charles Blundell, and Shane Legg. Noisy networks for exploration. In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=rywHCPkAW>. 3
- [25] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016. 3
- [26] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017. 3
- [27] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018. 3
- [28] Ian Osband, Benjamin Van Roy, Daniel J Russo, and Zheng Wen. Deep exploration via randomized value functions. *Journal of machine learning research*, 20(124): 1–62, 2019. 3
- [29] Ziyang Huang, Jiagang Chen, Jin Liu, and Shunping Ji. Opt3dgs: Optimizing 3d gaussian splatting with adaptive exploration and curvature-aware exploitation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 5230–5238, 2026. 3
- [30] Dmytro Kotovenko, Olga Grebenkova, and Björn Ommer. Edgs: Eliminating densification for efficient convergence of 3dgs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 41065–41076, 2026. 4
- [31] Lei Lan, Tianjia Shao, Zixuan Lu, Yu Zhang, Chenfanfu Jiang, and Yin Yang. 3dgs2: Near second-order converging 3d gaussian splatting. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pages 1–10, 2025. 4
- [32] He Zhu, Xiaotong Huang, Zihan Liu, Weikai Lin, Xiaohong Liu, Zhezhi He, Jingwen Leng, Minyi Guo, and Yu Feng. Seele: A unified acceleration framework for real-time gaussian splatting on mobile devices. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25979–25989, 2026. 6
- [33] Chengbo Wang, Guozheng Ma, Yifei Xue, and Yizhen Lao. Faster and better 3d splatting via group training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. 6, 20
- [34] Jiaqi Liu and Zhizhong Han. Speeding up the learning of 3d gaussians with much shorter gaussian lists. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1231–1240, 2026. 6
- [35] Baowen Zhang, Chuan Fang, Rakesh Shrestha, Yixun Liang, Xiao-Xiao Long, and Ping Tan. Rade-gs: Rasterizing depth in gaussian splatting. *ACM Transactions on Graphics*, 45(2):1–14, 2026. 6
- [36] Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 31(9):6100–6111, 2024. 6
- [37] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20310–20320, 2024. 9
- [38] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5521–5531, 2022. 9, 10
- [39] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in Neural Information Processing Systems*, 37:21875–21911, 2024. 10

Appendix

A The Prove of Orthogonality	16
A.1 Preliminaries	16
A.2 Gradient Decomposition	17
A.3 Prove of Orthogonality	18
B Factor-wise Dependence of the 2D Positional Gradient on Depth Sorting	19
B.1 Gradient Decomposition	20
B.2 m -Dependence of Each Factor	20
B.3 Derivation of Factor B $\partial \hat{\mathbf{c}}_c^{(p)} / \partial \alpha_m^{(p)}$	20
B.4 Gradient attenuation	21
C The Positional Perturbation Exploration	21

A. The Prove of Orthogonality

Goal. We prove that the 2D position path gradient $\mathbf{g}_{2d} = \partial L^{2D} / \partial \mathbf{P}_{3D}$ is always orthogonal to the viewing direction $\mathbf{D}_{C2G} = \mathbf{P}_{3D} - \mathbf{P}_{cam}$. This reveals a fundamental limitation: the 2D position updates alone provides **no gradient along the line of sight**. The prove proceeds by decomposing $\mathbf{g}_{2d}$ through the rendering pipeline via the chain rule and showing that the inner product $\mathbf{g}_{2d} \cdot \mathbf{D}_{C2G}$ collapses to zero through two geometric facts: (i) ${}^h\mathbf{p}$ lies in the null space of the perspective division Jacobian, and (ii) the viewport Jacobian annihilates the surviving z -component.

A.1. Preliminaries

Symbol Definition To rigorously derive the orthogonality property, we first establish the notation used throughout this prove. We denote the camera position in world coordinates as $\mathbf{P}_{cam}$, with pose $[\mathbf{R}, \mathbf{t}]$. For a Gaussian primitive G , let $\mathbf{P}_{3D}$ be its world-space position and $\mathbf{p}_{cam}$ its camera-space position. The rendering pipeline transforms $\mathbf{P}_{3D}$ through the following stages:

1. The view matrix $\mathbf{V}$ maps world coordinates to camera space: $[\mathbf{p}_{cam}^T, 1]^T = \mathbf{V}[\mathbf{P}_{3D}^T, 1]^T$.
2. The projection matrix $\mathbf{P}$ maps camera coordinates to homogeneous clip space: ${}^h\mathbf{p} = \mathbf{P}[\mathbf{p}_{cam}^T, 1]^T$.
3. Perspective division yields Normalized Device Coordinates (NDC): $\mathbf{p}_{ndc} = [\frac{p_x}{p_w}, \frac{p_y}{p_w}, \frac{p_z}{p_w}]^T$.
4. The viewport transformation with image dimensions $W \times H$ produces 2D pixel coordinates $\mathbf{p}_{2D}$.

The composite matrix $\mathbf{M} = \mathbf{PV}$ encodes the combined camera–projection mapping. These symbols are summarized below.

$$\begin{aligned}
 \mathbf{P}_{3D} &= [x, y, z]^T \\
 \mathbf{p}_{\text{cam}} &= [p_x^G, p_y^G, p_z^G]^T \\
 {}^h\mathbf{p} &= [p_x, p_y, p_z, p_w]^T \\
 \mathbf{p}_{\text{ndc}} &= [x_n, y_n, z_n]^T \\
 \mathbf{p}_{2D} &= [u, v]^T \\
 \mathbf{M} &= \mathbf{P}\mathbf{V} \\
 \mathbf{V} &= \begin{bmatrix} R_{00} & R_{01} & R_{02} & t_x \\ R_{10} & R_{11} & R_{12} & t_y \\ R_{20} & R_{21} & R_{22} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
 \mathbf{P} &= \begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}
 \end{aligned} \tag{4}$$

Forward Mapping The forward mapping proceeds through four stages:

World to camera.

$$[\mathbf{p}_{\text{cam}}^T, 1]^T = [p_x^G, p_y^G, p_z^G, 1]^T = \mathbf{V} [\mathbf{P}_{3D}^T, 1]^T, \tag{5}$$

Camera to clip.

$${}^h\mathbf{p} = [p_x, p_y, p_z, p_w]^T = \mathbf{M} [\mathbf{P}_{3D}^T, 1]^T = \mathbf{P} [\mathbf{p}_{\text{cam}}^T, 1]^T, \tag{6}$$

Clip to NDC.

$$\mathbf{p}_{\text{ndc}} = \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = \begin{bmatrix} p_x/p_w \\ p_y/p_w \\ p_z/p_w \end{bmatrix}, \tag{7}$$

NDC to pixel.

$$\mathbf{p}_{2D} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} (x_n + 1) \cdot \frac{W}{2} \\ (y_n + 1) \cdot \frac{H}{2} \end{bmatrix}. \tag{8}$$

A.2. Gradient Decomposition

The gradient of the loss with respect to the 3D Gaussian position comprises three terms:

$$\frac{\partial L}{\partial \mathbf{P}_{3D}} = \frac{\partial L}{\partial \mathbf{Cov2D}} \frac{\partial \mathbf{Cov2D}}{\partial \mathbf{P}_{3D}} + \frac{\partial L}{\partial \mathbf{p}_{2D}} \frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{P}_{3D}} + \frac{\partial L}{\partial \mathbf{SH}} \frac{\partial \mathbf{SH}}{\partial \mathbf{P}_{3D}}. \tag{9}$$

These correspond to the optimization paths through the 2D covariance, the 2D screen-space position, and the spherical harmonics coefficients, respectively. Among them, the 2D position term dominates in magnitude; we therefore focus on this path and decompose it via the chain rule into three Jacobian matrices.

The 2D position gradient further decomposes via the chain rule through the rendering pipeline:

$$\mathbf{g}_{2d} = \frac{\partial L^{2D}}{\partial \mathbf{P}_{3D}} = \frac{\partial L}{\partial \mathbf{p}_{2D}} \frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{p}_{\text{ndc}}} \frac{\partial \mathbf{p}_{\text{ndc}}}{\partial {}^h\mathbf{p}} \frac{\partial {}^h\mathbf{p}}{\partial \mathbf{P}_{3D}} \tag{10}$$

The core factor $\frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{P}_{3D}}$ is the product of three Jacobian matrices, which we now derive explicitly.

$$\frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{p}_{ndc}} = \begin{bmatrix} \frac{\partial u}{\partial x_n} & \frac{\partial u}{\partial y_n} & \frac{\partial u}{\partial z_n} \\ \frac{\partial v}{\partial x_n} & \frac{\partial v}{\partial y_n} & \frac{\partial v}{\partial z_n} \end{bmatrix} = \begin{bmatrix} \frac{W}{2} & 0 & 0 \\ 0 & \frac{H}{2} & 0 \end{bmatrix}. \quad (11)$$

$$\begin{aligned} \frac{\partial \mathbf{p}_{ndc}}{\partial^h \mathbf{p}} &= \begin{bmatrix} \frac{\partial x_n}{\partial p_x} & \frac{\partial x_n}{\partial p_y} & \frac{\partial x_n}{\partial p_z} & \frac{\partial x_n}{\partial p_w} \\ \frac{\partial y_n}{\partial p_x} & \frac{\partial y_n}{\partial p_y} & \frac{\partial y_n}{\partial p_z} & \frac{\partial y_n}{\partial p_w} \\ \frac{\partial z_n}{\partial p_x} & \frac{\partial z_n}{\partial p_y} & \frac{\partial z_n}{\partial p_z} & \frac{\partial z_n}{\partial p_w} \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{p_w} & 0 & 0 & -\frac{p_x}{p_w^2} \\ 0 & \frac{1}{p_w} & 0 & -\frac{p_y}{p_w^2} \\ 0 & 0 & \frac{1}{p_w} & -\frac{p_z}{p_w^2} \end{bmatrix}. \end{aligned} \quad (12)$$

$$\begin{aligned} \frac{\partial^h \mathbf{p}}{\partial \mathbf{P}_{3D}} &= \begin{bmatrix} \frac{\partial p_x}{\partial x} & \frac{\partial p_x}{\partial y} & \frac{\partial p_x}{\partial z} \\ \frac{\partial p_y}{\partial x} & \frac{\partial p_y}{\partial y} & \frac{\partial p_y}{\partial z} \\ \frac{\partial p_z}{\partial x} & \frac{\partial p_z}{\partial y} & \frac{\partial p_z}{\partial z} \end{bmatrix} \\ &= \begin{bmatrix} M_{00} & M_{01} & M_{02} \\ M_{10} & M_{11} & M_{12} \\ M_{20} & M_{21} & M_{22} \\ M_{30} & M_{31} & M_{32} \end{bmatrix} \\ &= \mathbf{M}_{[:,3]} = \mathbf{P}\mathbf{V}_{[:,3]}. \end{aligned} \quad (13)$$

For the orthogonality prove we also need the world-space direction from the camera to the Gaussian, $\mathbf{P}_{3D} - \mathbf{P}_{cam}$. Applying $\mathbf{V}_{[:,3]}$ to this direction and then $\mathbf{P}$ yields:

$$\mathbf{P}\mathbf{V}_{[:,3]}(\mathbf{P}_{3D} - \mathbf{P}_{cam}) = \mathbf{P} [\mathbf{p}_{cam}^T, 0]^T, \quad (14)$$

which, after perspective division, gives the vanishing point of all lines parallel to the camera-to-Gaussian ray $\mathbf{p}_{cam}$ in camera space.

A.3. Prove of Orthogonality

We prove that $\mathbf{g}_{2d}$ is always orthogonal to the viewing direction from the camera center to the primitive. This orthogonality property is formalized in [Equation 16](#).

Let $\mathbf{D}_{C2G} = \mathbf{P}_{3D} - \mathbf{P}_{cam}$ be the viewing direction from the camera to the Gaussian. Using the gradient decomposition

from Equation 10, their inner product expands as

$$\begin{aligned}
 \frac{\partial L^{2D}}{\partial \mathbf{P}_{3D}} \mathbf{D}_{C2G} &= \frac{\partial L^{2D}}{\partial \mathbf{P}_{3D}} (\mathbf{P}_{3D} - \mathbf{P}_{\text{cam}}) \\
 &= \frac{\partial L}{\partial \mathbf{p}_{2D}} \frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{p}_{\text{ndc}}} \frac{\partial \mathbf{p}_{\text{ndc}}}{\partial {}^h \mathbf{p}} \frac{\partial {}^h \mathbf{p}}{\partial \mathbf{P}_{3D}} (\mathbf{P}_{3D} - \mathbf{P}_{\text{cam}}) \\
 &= \frac{\partial L}{\partial \mathbf{p}_{2D}} \frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{p}_{\text{ndc}}} \frac{\partial \mathbf{p}_{\text{ndc}}}{\partial {}^h \mathbf{p}} \mathbf{P} \mathbf{V}_{[:,3]} (\mathbf{P}_{3D} - \mathbf{P}_{\text{cam}}) \\
 &= \frac{\partial L}{\partial \mathbf{p}_{2D}} \frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{p}_{\text{ndc}}} \frac{\partial \mathbf{p}_{\text{ndc}}}{\partial {}^h \mathbf{p}} \mathbf{P} [\mathbf{p}_{\text{cam}}^T, 0]^T \\
 &= \frac{\partial L}{\partial \mathbf{p}_{2D}} \frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{p}_{\text{ndc}}} \frac{\partial \mathbf{p}_{\text{ndc}}}{\partial {}^h \mathbf{p}} \begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} p_x^G \\ p_y^G \\ p_z^G \\ 0 \end{bmatrix} \\
 &= \frac{\partial L}{\partial \mathbf{p}_{2D}} \frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{p}_{\text{ndc}}} \frac{\partial \mathbf{p}_{\text{ndc}}}{\partial {}^h \mathbf{p}} \begin{bmatrix} \frac{2n}{r-l} p_x^G + \frac{r+l}{r-l} p_z^G \\ \frac{2n}{t-b} p_y^G + \frac{t+b}{t-b} p_z^G \\ -\frac{f+n}{f-n} p_z^G \\ -p_z^G \end{bmatrix} \\
 &= \frac{\partial L}{\partial \mathbf{p}_{2D}} \frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{p}_{\text{ndc}}} \frac{\partial \mathbf{p}_{\text{ndc}}}{\partial {}^h \mathbf{p}} \begin{bmatrix} p_x \\ p_y \\ p_z + \frac{2fn}{f-n} \\ p_w \end{bmatrix} \\
 &= \frac{\partial L}{\partial \mathbf{p}_{2D}} \frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{p}_{\text{ndc}}} \begin{bmatrix} \frac{1}{p_w} & 0 & 0 & -\frac{p_x}{p_w^2} \\ 0 & \frac{1}{p_w} & 0 & -\frac{p_y}{p_w^2} \\ 0 & 0 & \frac{1}{p_w} & -\frac{p_z}{p_w^2} \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z + \frac{2fn}{f-n} \\ p_w \end{bmatrix} \\
 &= \frac{\partial L}{\partial \mathbf{p}_{2D}} \begin{bmatrix} \frac{W}{2} & 0 & 0 \\ 0 & \frac{H}{2} & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \frac{2fn}{(f-n)p_w} \end{bmatrix} \\
 &= \frac{\partial L}{\partial \mathbf{p}_{2D}} \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\
 &= 0 \\
 \frac{\partial L^{2D}}{\partial \mathbf{P}_{3D}} \mathbf{D}_{C2G} &= \frac{\partial L^{2D}}{\partial \mathbf{P}_{3D}} (\mathbf{P}_{3D} - \mathbf{P}_{\text{cam}}) = 0
 \end{aligned} \tag{15}$$

The derivation above reveals two geometric reasons behind this orthogonality. First, since $\mathbf{p}_{\text{ndc}}$ is homogeneous of degree zero in ${}^h \mathbf{p}$, the homogeneous coordinate vector ${}^h \mathbf{p}$ lies in the null space of $\frac{\partial \mathbf{p}_{\text{ndc}}}{\partial {}^h \mathbf{p}}$. Combined with $\mathbf{P}[\mathbf{p}_{\text{cam}}^T, 0]^T = {}^h \mathbf{p} + [0, 0, \frac{2fn}{f-n}, 0]^T$, this forces the intermediate vector after perspective division to carry only a z -component, i.e., $[0, 0, \frac{2fn}{(f-n)p_w}]^T$. Second, left-multiplying by the viewport Jacobian $\frac{\partial \mathbf{p}_{2D}}{\partial \mathbf{p}_{\text{ndc}}}$, whose third column is identically zero, annihilates this z -component and yields $[0, 0]^T$. Multiplying by $\frac{\partial L}{\partial \mathbf{p}_{2D}}$ then trivially gives zero. Consequently, $\mathbf{D}_{C2G}$ is always orthogonal to $\frac{\partial L^{2D}}{\partial \mathbf{P}_{3D}}$. **In essence, the orthogonality arises from two geometric facts: ${}^h \mathbf{p}$ lies in the null space of the perspective division Jacobian, forcing the intermediate vector to a pure z -component; the viewport Jacobian then annihilates this component.** $\square$

B. Factor-wise Dependence of the 2D Positional Gradient on Depth Sorting

Goal. We prove that along a pixel’s ray, the 2D positional gradient magnitude $|\partial L / \partial x_m|$ of a depth-sorted Gaussian G_m decays rapidly as the sorted index m increases, making rear Gaussians increasingly unable to surpass the densification threshold τ . This decay is driven by the accumulated transmittance $T_{m-1} = \prod_{j=1}^{m-1} (1 - \alpha_j)$ of all front Gaussians: T_{m-1} is monotonically non-increasing in m , so the gradient bound compresses toward zero—the direct cause of **blending-induced densification failure**. We formalize this by decomposing the gradient into factors, identifying the sole m -dependent factor, and bounding its

magnitude by T_{m-1} .

B.1. Gradient Decomposition

For a pixel p , let the depth-sorted Gaussians along its ray be $G_1, \dots, G_N$. The rendered color follows the α -blending equation:

$$\begin{aligned}\hat{\mathbf{c}}^{(p)} &= \sum_{i=1}^N \mathbf{c}_i \alpha_i^{(p)} T_{i-1}^{(p)} + \mathbf{c}_{\text{bg}} T_N^{(p)}, \\ T_k^{(p)} &= \prod_{j=1}^k (1 - \alpha_j^{(p)}),\end{aligned}\tag{17}$$

where $\alpha_i^{(p)} = o_i \cdot G_i^{2D}(p)$ is the effective opacity of G_i at pixel p , with learned opacity $o_i \in (0, 1)$ and 2D Gaussian $G_i^{2D}(p) \in (0, 1]$; colors satisfy $\mathbf{c}_i, \mathbf{c}_{\text{bg}} \in [0, 1]^3$.

We decompose $\partial L / \partial x_m$ via the chain rule:

$$\frac{\partial L}{\partial x_m} = \sum_p \sum_{c=1}^C \underbrace{\frac{\partial L}{\partial \hat{\mathbf{c}}_c^{(p)}}}_{\text{Factor A}} \cdot \underbrace{\frac{\partial \hat{\mathbf{c}}_c^{(p)}}{\partial \alpha_m^{(p)}}}_{\text{Factor B}} \cdot \underbrace{o_m}_{\text{Factor C}} \cdot \underbrace{\frac{\partial G_m^{2D, (p)}}{\partial x_m}}_{\text{Factor D}},\tag{18}$$

and identically for $\partial L / \partial y_m$ with $\partial G_m^{2D, (p)} / \partial y_m$ in Factor D.

B.2. m -Dependence of Each Factor

We examine which factors in Equation 18 are influenced by the depth-sorted index m .

Factor A ($\partial L / \partial \hat{\mathbf{c}}_c^{(p)}$): *independent of m* . This term is the derivative of the loss with respect to the rendered pixel color; it does not involve m and is therefore independent of depth ordering.

Factor C (o_m): *independent of m* . The opacity is a learned intrinsic parameter of Gaussian G_m . Its value is attached to the primitive itself and does not change when the view-dependent sorting reassigns which Gaussian occupies slot m .

Factor D ($\partial G_m^{2D, (p)} / \partial x_m$): *independent of m* . This spatial derivative of the 2D Gaussian kernel depends only on the screen-space covariance Σ_m and the pixel offset $\Delta^{(p)} = [x_m - x_p, y_m - y_p]^T$ —quantities determined by the intrinsic 3D position and covariance of G_m , neither of which is affected by the depth-sorted ordering.

Factor B ($\partial \hat{\mathbf{c}}_c^{(p)} / \partial \alpha_m^{(p)}$): *depends on m* . This factor couples G_m to every other Gaussian through the α -blending equation. It is the sole channel through which the depth-sorted index m enters the gradient. We now derive its explicit form.

B.3. Derivation of Factor B $\partial \hat{\mathbf{c}}_c^{(p)} / \partial \alpha_m^{(p)}$

Dropping the pixel superscript (p) for brevity, we split Equation 17 at index m to isolate terms involving α_m :

$$\hat{\mathbf{c}} = \underbrace{\sum_{i=1}^{m-1} \mathbf{c}_i \alpha_i T_{i-1}}_{\text{in front of } G_m} + \underbrace{\mathbf{c}_m \alpha_m T_{m-1}}_{G_m \text{ itself}} + \underbrace{\sum_{i=m+1}^N \mathbf{c}_i \alpha_i T_{i-1} + \mathbf{c}_{\text{bg}} T_N}_{\text{behind } G_m}.\tag{19}$$

Differentiating Equation 19 w.r.t. α_m [33]:

$$\begin{aligned}\frac{\partial \hat{\mathbf{c}}}{\partial \alpha_m} &= T_{m-1} \left[\mathbf{c}_m - \underbrace{\left(\sum_{i=m+1}^N \mathbf{c}_i \alpha_i \prod_{j=m+1}^{i-1} (1 - \alpha_j) + \mathbf{c}_{\text{bg}} \prod_{j=m+1}^N (1 - \alpha_j) \right)}_{= \hat{\mathbf{c}}_{m+1:N} \in [0, 1]^3} \right] \\ &= T_{m-1} \cdot (\mathbf{c}_m - \hat{\mathbf{c}}_{m+1:N}),\end{aligned}\tag{20}$$

where $\hat{\mathbf{c}}_{m+1:N}$ denotes the alpha-composited color of the Gaussians behind G_m over the background.

Since both $\mathbf{c}_m$ and $\hat{\mathbf{c}}_{m+1:N}$ lie in $[0, 1]^3$, their per-channel difference is bounded by 1, yielding the bound

$$\left| \frac{\partial \hat{\mathbf{c}}_c}{\partial \alpha_m} \right| \leq T_{m-1}, \quad \forall c \in \{1, 2, 3\}, \quad (21)$$

where $T_{m-1} = \prod_{j=1}^{m-1} (1 - \alpha_j)$ is the accumulated transmittance of the $m - 1$ Gaussians in front of G_m .

Monotonicity of the bound. Each $\alpha_j \in [0, 1]$ implies $1 - \alpha_j \in [0, 1]$, hence $T_k = T_{k-1} \cdot (1 - \alpha_k) \leq T_{k-1}$ for all $k \geq 1$. The transmittance sequence is therefore monotonically non-increasing:

$$T_0 = 1 \geq T_1 \geq T_2 \geq \dots \geq T_N \geq 0. \quad (22)$$

Consequently, the upper bounds in Equation 21 can only decrease or stay constant as the depth-sorted index m grows.

B.4. Gradient attenuation

The exact gradient $T_{m-1} \cdot (\mathbf{c}_m - \hat{\mathbf{c}}_{m+1:N})$ depends on m through two factors: the transmittance T_{m-1} , which decays monotonically, and the color difference $\mathbf{c}_m - \hat{\mathbf{c}}_{m+1:N}$. The latter varies non-monotonically with m , because advancing m alters both which Gaussian occupies slot m and the rear subsequence $\hat{\mathbf{c}}_{m+1:N}$ being composited.

While the color-difference term may cause the gradient magnitude to fluctuate, the magnitude can never exceed the monotonic envelope T_{m-1} , and this envelope decays monotonically toward zero as front Gaussians accumulate. This attenuation is inescapable: T_{m-1} depends only on the Gaussians in front of position m , so the gradient bound lies entirely outside the control of rear Gaussians.

As m increases, rear Gaussians therefore receive a progressively weaker 2D positional gradient signal, making them increasingly unable to surpass the densification threshold τ —the direct mechanism of blending-induced densification failure. $\square$

C. The Positional Perturbation Exploration



Figure 14: Positional perturbation exploration along the viewing ray.

We adapt the positional perturbation mechanism [8] from 3DGS-MCMC to the standard 3DGS pipeline, constraining perturbation direction strictly along the viewing ray. The results demonstrate that even this straightforward depth-directed intervention significantly enhances reconstruction fidelity in unbounded scenes, as shown in Figure 14.