ANISOTROPY-GATED DENSIFICATION AGAINST POISON-SPLAT ATTACKS

Anonymous authors
Paper under double-blind review

ABSTRACT

3D Gaussian Splatting (3DGS) achieves efficient novel-view synthesis through explicit Gaussian representations and Adaptive Density Control (ADC). However, recent Poison-splat attacks exploit the gradient-driven densification mechanism to trigger excessive clone and split operations, leading to rapid Gaussian proliferation and substantial computational overhead. We observe that poisoned scenes exhibit stronger spectral anisotropy than clean scenes, suggesting that frequency-domain characteristics provide useful cues for regulating densification. Based on this observation, we propose Anisotropy-Gated Densification (AGD), a defense method that integrates spectral anisotropy into ADC. AGD adaptively adjusts the densification threshold according to scene-level spectral statistics and guides gradient accumulation using local spectral orientation, suppressing unnecessary Gaussian generation while preserving legitimate scene structures. Extensive experiments on Mip-NeRF360, NeRF Synthetic, and Tanks and Temples demonstrate that AGD effectively mitigates Poison-splat attacks by reducing Gaussian proliferation, GPU memory consumption, and training overhead, while maintaining rendering quality comparable to the original 3DGS on clean scenes.

1 INTRODUCTION

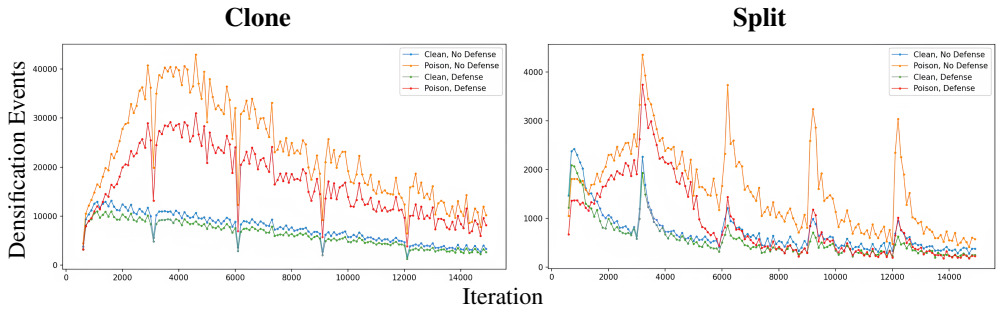
3D Gaussian Splatting (3DGS) enables efficient real-time novel-view synthesis using explicit Gaussian primitives and differentiable rasterization (Kerbl et al., 2023). Unlike implicit neural representations (Mildenhall et al., 2021), 3DGS directly optimizes Gaussian primitives, achieving fast training and rendering while maintaining high reconstruction quality. A key component is Adaptive Density Control (ADC), which dynamically clones or splits Gaussian primitives according to accumulated view-space positional gradients. However, this gradient-driven mechanism introduces a vulnerability. Since ADC primarily relies on gradient magnitude to determine densification, it cannot distinguish legitimate optimization signals from adversarially amplified ones. Poison-splat (Lu et al., 2025) exploits this weakness by injecting imperceptible perturbations into training images, repeatedly triggering clone and split operations and thereby increasing training cost. As shown in Fig. 1, poisoned scenes exhibit substantially more densification events than clean scenes from the early stages of training.

Existing defenses (Bui-Huynh et al., 2026) mainly address poisoned inputs (Li et al., 2026; Qiao et al., 2026b), modify the optimization process (Chen et al., 2026), or operate on the generated representation, rather than directly regulating the densification decision. Motivated by this observation, we propose Anisotropy-Gated Densification (AGD), a defense that incorporates frequency-domain structural information into ADC. AGD combines scene-level threshold adaptation and direction-aware gradient accumulation based on spectral anisotropy and orientation, together with late-stage scale regularization to suppress unnecessary Gaussian generation while preserving reconstruction quality.

The main contributions are summarized as follows:

- **Empirical Characterization of Poison-induced Densification.** Our empirical analysis shows that Poison-splat substantially amplifies the gradient-driven densification behavior of ADC, causing excessive clone and split operations from the early stages of optimization.

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.



Tanks and Temples – Auditorium

Figure 1: Comparison of densification events on Auditorium of Tanks and Temples. Without defense, poisoned inputs induce substantially more densification events than clean inputs. Our defense (AGD) substantially reduces these events under poisoned training, while leaving densification on clean inputs largely unchanged, suggesting that AGD suppresses attack-induced proliferation while largely preserving densification behavior on clean inputs.

- **Anisotropy-Gated Densification (AGD).** We propose a defense that incorporates spectral information into densification through scene-level threshold adaptation and direction-aware gradient accumulation, together with scale regularization for stabilizing Gaussian geometry.

- **Comprehensive Evaluation.** Experiments on Mip-NeRF360 (MIP) (Barron et al., 2022), NeRF Synthetic (NS) (Mildenhall et al., 2021), and Tanks and Temples (TT) (Knapitsch et al., 2017) show that AGD suppresses Gaussian proliferation and reduces GPU memory consumption and training overhead under Poison-splat while preserving comparable rendering quality.

2 Related Work

2.1 3D Gaussian Splatting and Adaptive Density Control

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) represents a scene with explicit Gaussian primitives and uses differentiable rasterization for efficient novel-view synthesis. Its representation has been extended to dynamic reconstruction (Wu et al., 2024; Huang et al., 2024), alias-free rendering (Yu et al., 2024), scene editing (Yan et al., 2024; Ye et al., 2024a), sparse-view reconstruction (Park et al., 2025; Li et al., 2024), efficient rendering (Niemeyer et al., 2025; Fang & Wang, 2024), compression and pruning (Fan et al., 2024; Hanson et al., 2025b; Lee et al., 2026; Hanson et al., 2025a; Liu et al., 2024), and structured representations (Lu et al., 2024; Ververas et al., 2024).

Adaptive Density Control (ADC) is central to 3DGS: it adjusts the number and placement of Gaussians during optimization. In the original formulation, view-space positional gradients are accumulated, and Gaussians whose accumulated gradients exceed a threshold are cloned or split. Subsequent work has refined gradient statistics (Rota Bulò et al., 2024; Deng et al., 2026), incorporated color- or pixel-aware signals (Kim et al., 2024; Zhang et al., 2024), used residual- or structure-aware guidance (Lyu et al., 2025; Ververas et al., 2024; Lyu et al., 2026), and investigated directional or alternative gradient criteria (Ye et al., 2024b; Jeong et al., 2026; Deng et al., 2024; Zhou et al., 2026; Wang et al., 2025; Grubert et al., 2025).

These methods focus on deciding where additional Gaussians are needed based on signals observed during training. Their reliance on those signals raises a distinct robustness question: can adversarial perturbations distort the densification decisions and, consequently, the resulting Gaussian representation?

2.2 Security Threats on 3D Gaussian Splatting

The explicit representation and optimization pipeline of 3DGS introduce new security concerns, including ownership protection (Huang et al., 2026) and adversarial manipulation. Existing attacks such as 3DGAA (Zhang et al., 2026), StealthAttack (Ke et al., 2025), GaussTrap (Hong et al., 2025), and AdvSplat (Qiao et al., 2026a) investigate different forms of adversarial or poisoning attacks against Gaussian-based systems.

Among them, Poison-splat (Lu et al., 2025) specifically targets the computational behavior of 3DGS by optimizing bounded perturbations that amplify the view-space positional gradients used by ADC. These manipulated gradients repeatedly trigger clone and split operations, causing excessive Gaussian generation, increased GPU memory consumption, and longer training time.

2.3 Defenses Against Poison-splat Attacks

Existing defenses act at the input or optimization stages. RemedyGS (Li et al., 2026) detects and purifies poisoned images before reconstruction, and DefenseSplat (Qiao et al., 2026b) uses frequency-aware processing to mitigate poisoned inputs. Spectral Defense (Chen et al., 2026) introduces the frequency filter and frequency-domain constraints during optimization. Poison-splat, however, increases computational cost through ADC's decision to create new Gaussians. This suggests a complementary intervention at the densification step: determining whether a high gradient reflects a genuine need for additional primitives before allowing cloning or splitting. This motivates treating the densification process itself as a direct intervention point for suppressing unnecessary Gaussian generation.

3 Method

Poison-splat attacks (Lu et al., 2025) exploit the Adaptive Density Control (ADC) mechanism of 3DGS (Kerbl et al., 2023) by manipulating the gradient signals used for Gaussian densification. Our analysis shows that poisoned scenes trigger excessive clone and split operations from the early stages of optimization, increasing training cost. Since this overhead originates from densification, we propose Anisotropy-Gated Densification (AGD), a framework that regulates ADC using spectral information extracted from the input images.

AGD incorporates scene- and tile-level spectral cues into gradient-based densification. It first estimates scene-level spectral anisotropy to initialize the densification threshold. During optimization, tile-level anisotropy and dominant spectral orientation guide gradient accumulation, emphasizing signals consistent with local spectral structures. A separate scale regularization term is applied during later optimization to stabilize Gaussian geometry.

3.1 Preliminary

3D Gaussian Splatting (3DGS). 3DGS represents a scene using anisotropic 3D Gaussian primitives. Given a spatial location x, each Gaussian G_i is parameterized by a center μ_i and covariance matrix

$$\Sigma_i = R_i S_i S_i^T R_i^T, \tag{1}$$

where R_i and S_i denote the rotation and scaling matrices, respectively. Each primitive additionally stores opacity and view-dependent color represented by Spherical Harmonics. During rendering, Gaussians are projected onto the image plane and composited through differentiable α-blending.

Adaptive Density Control (ADC). 3DGS initializes Gaussian primitives from a sparse Structure-from-Motion (SfM) point cloud. During optimization, ADC accumulates view-space positional gradients and densifies Gaussians whose accumulated gradient magnitude exceeds a threshold τ. Depending on their scale, Gaussians are cloned or split into smaller primitives. Thus, ADC directly relies on gradient magnitude to determine where additional representation capacity is required.

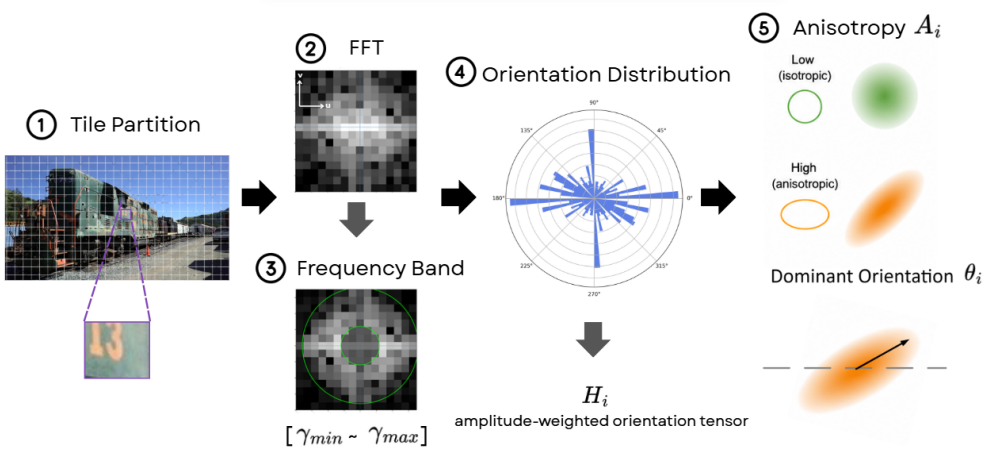


Figure 2: Illustration of the proposed tile-level spectral descriptors. Each image tile is transformed into the frequency domain using 2D FFT, followed by radial frequency selection and construction of a spectral orientation tensor. Eigen-decomposition yields the spectral anisotropy score A_i and dominant orientation θ_i, characterizing the strength and direction of local frequency anisotropy.

3.2 Tile-level Spectral Anisotropy Measurement

As illustrated in Fig. 2, we extract a spectral descriptor that characterizes the directional distribution of local frequency components. We partition an input image I into N non-overlapping tiles $\{T_i\}_{i=1}^{N}$. For each tile, we compute the two-dimensional discrete Fourier transform and its magnitude:

$$F_i(u, v) = \mathrm{FFT2}(T_i), \qquad M_i(u, v) = |F_i(u, v)|. \tag{2}$$

We retain coefficients whose normalized radial frequencies lie in the intermediate band $[\gamma_{\min}, \gamma_{\max}]$. Let $\mathcal{K}_i$ denote the retained coefficients for tile i. For each $k \in \mathcal{K}_i$, let m_k be its magnitude and θ_k its orientation in the frequency plane. We form the amplitude-weighted orientation tensor

$$H_i = \frac{1}{S_i} \sum_{k \in \mathcal{K}_i} m_k \begin{bmatrix} \cos^2 \theta_k & \cos \theta_k \sin \theta_k \\ \cos \theta_k \sin \theta_k & \sin^2 \theta_k \end{bmatrix}, \qquad S_i = \sum_{k \in \mathcal{K}_i} m_k. \tag{3}$$

For $S_i = 0$, we set $H_i = \mathbf{0}$. When $S_i > 0$, the tensor has unit trace: coefficients concentrated along one frequency-plane direction produce a dominant eigenvalue, whereas a uniform orientation distribution produces approximately equal eigenvalues. Let $\lambda_{i,1} \geq \lambda_{i,2}$ be the eigenvalues of H_i. We define tile-level spectral anisotropy as

$$A_i = \lambda_{i,1} - \lambda_{i,2}, \tag{4}$$

with larger values indicating stronger directional concentration. The principal eigenvector gives the dominant *frequency-plane* orientation for tile i. Finally, we aggregate the tile-level scores using the median:

$$A(I) = \mathrm{median}_{i=1,\ldots,N} A_i. \tag{5}$$

The median summarizes local spectral anisotropy while limiting the influence of individual tiles.

3.3 Anisotropy-Gated Densification

To regulate gradient-driven densification without modifying the underlying clone and split operations, we propose Anisotropy-Gated Densification (AGD), which augments ADC with spectral information extracted from the training images, as illustrated in Fig. 3. AGD regulates densification from two complementary perspectives: scene-level threshold adaptation and tile-level direction-aware gradient accumulation. A separate scale regularization term is introduced later to stabilize Gaussian geometry.

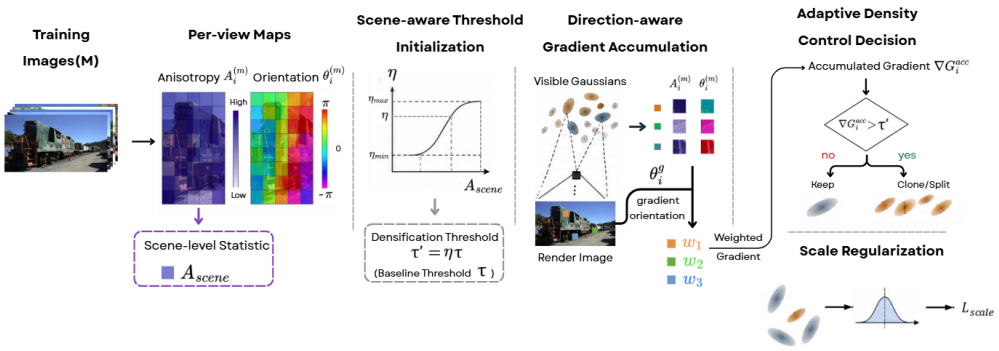


Figure 3: Overall pipeline of AGD. Per-view anisotropy $A_i^{(m)}$ and orientation $\theta_i^{(m)}$ maps are extracted from M training images to (i) adaptively initialize the densification threshold τ' based on the scene-level statistic A_{scene} , and (ii) weight gradient accumulation according to directional consistency, producing an anisotropy-aware accumulated gradient g_i^{acc} for the ADC decision. AGD additionally incorporates a scale regularization term L_{scale} (detailed in Sec. 3.4) that discourages the formation of needle-like Gaussian configurations.

3.3.1 Scene-aware Threshold Initialization (STI)

The first component regulates the overall sensitivity of densification before optimization begins. Instead of applying a fixed threshold to all scenes, AGD adapts the threshold according to the global spectral characteristics of the training data. For each training image, the tile-level spectral anisotropy described in Sec. 3.2 is aggregated via Eq. 5 to obtain an image-level score. Given M training images, the scene-level anisotropy is defined as

$$A_{scene} = \text{median}_{m=1,\dots,M} A^{(m)}. \tag{6}$$

We convert this score into a threshold scaling factor using a logistic mapping:

$$\eta = 1 + (\eta_{max} - 1)\sigma(\frac{A_{scene} - A_T}{\delta}), \tag{7}$$

where A_T denotes the anisotropy baseline of the corresponding dataset, δ controls the transition smoothness, η_{max} limits the maximum scaling ratio, and $\sigma(\cdot)$ denotes the sigmoid function. The original ADC threshold τ is then adjusted as

$$\tau' = \eta\tau. \tag{8}$$

The scaling factor increases smoothly with scene-level anisotropy, reducing the sensitivity of densification to unusually large gradients. Since STI is computed once before training, its additional computational cost is negligible.

3.3.2 Direction-aware Gradient Accumulation (DGA)

While STI regulates the global densification sensitivity, DGA incorporates local spectral orientation into gradient accumulation. Before training, each image is divided into fixed-size tiles, and the spectral analysis (see Sec. 3.2) provides an anisotropy magnitude $A_i^{(m)}$ and dominant orientation $\theta_i^{(m)}$ for each tile. These descriptors are stored as per-camera direction maps and remain fixed during optimization.

At each iteration, a visible Gaussian is projected onto the image plane, and its projected location is used to retrieve the corresponding local anisotropy and spectral orientation. Let the 2D image-space positional gradient of Gaussian G_i be

$$\nabla G_i = (g_i^x, g_i^y), \tag{9}$$

which is obtained from the rendering process, and its orientation is computed as

$$\theta_i^g = Atan2(g_i^y, g_i^x). \tag{10}$$

To account for the π-periodicity of orientation, the directional consistency between the gradient and local spectral orientation is measured as

$$c_i = Cos(2(\theta_i^g - \theta_i)), \tag{11}$$

After combining with the local anisotropy, the accumulation weight is obtained as follows:

$$w_i = 1 + \alpha A_i c_i' \text{ and } c_i' = \frac{c_i + 1}{2}, \tag{12}$$

where A_i determines the confidence assigned to the local orientation and α determines the strength of spectral guidance, with a default value of 1. The weight is further constrained to make $w_i \in [w_{min}, w_{max}]$ to prevent unstable gradient amplification.

Finally, the weighted gradient is incorporated into the original ADC accumulation:

$$\nabla G_i^{acc} \leftarrow \nabla G_i^{acc} + w_i \nabla G_i, \tag{13}$$

where ∇G_i^{acc} denotes the accumulated positional gradient of the i-th Gaussian. Thus, gradients that are directionally consistent with local spectral structures receive greater influence during accumulation, while inconsistent gradients are prevented from receiving the same amplification. The subsequent clone and split decisions remain unchanged, but are made using the spectrally guided accumulated gradients.

3.4 Scale Regularization (SR)

AGD also includes a regularizer for needle-like Gaussian configurations. Controlled deletion experiments on trained clean scenes (Fig. 4) suggest that such Gaussians have relatively small marginal contributions to rendering quality. We therefore target needle-like shapes rather than penalizing scale anisotropy in general.

Let $s_i = (s_{i,x}, s_{i,y}, s_{i,z})$ denote the three principal-axis scales of Gaussian G_i, and define

$$\mu_i = \frac{1}{3} \sum_{d \in \{x,y,z\}} s_{i,d}, \qquad v_i = \frac{1}{3} \sum_{d \in \{x,y,z\}} (s_{i,d} - \mu_i)^2. \tag{14}$$

Following DefenseSplat (Qiao et al., 2026b), we measure relative scale dispersion as

$$\nu_i = \frac{v_i}{\mu_i^2 + \epsilon}. \tag{15}$$

We further compute the standardized third central moment:

$$\gamma_i = \frac{\frac{1}{3} \sum_{d \in \{x,y,z\}} (s_{i,d} - \mu_i)^3}{v_i^{3/2} + \epsilon}. \tag{16}$$

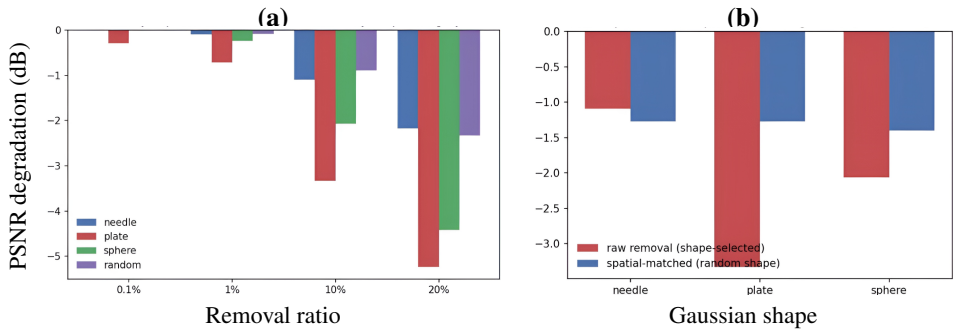
While ν_i measures the degree of scale imbalance, positive γ_i helps identify configurations with one dominant axis, and ϵ is a small constant for numerical stability. Their thresholded combination defines

$$\mathcal{L}_{\text{scale}} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} [\nu_i - \tau_\nu]_+ [\gamma_i - \tau_\gamma]_+, \tag{17}$$

where $[z]_+ = \max(0, z)$, $\mathcal{V}$ is the set of visible Gaussians, and τ_ν and τ_γ are thresholds. The penalty is nonzero only when both scale dispersion and positive skewness exceed their thresholds. At iteration t, the objective is

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \lambda_s \mathbb{I}[t \geq 5000] \, \mathcal{L}_{\text{scale}}, \tag{18}$$

where $\mathcal{L}_{\text{recon}}$ is the original 3DGS reconstruction objective and λ_s controls regularization strength. To preserve early-stage reconstruction and densification flexibility, $\mathcal{L}_{scale}$ is activated from iteration 5000 onward.



Tanks and Temples – Auditorium

Figure 4: Rendering impact of Gaussian shape removal: (a) PSNR degradation under different removal ratios for needle-like, plate-like, spherical, and randomly selected Gaussians; (b) comparison between 10% Gaussian raw removal (shape-selected) and spatial-matched removal (random shape), where the latter selects Gaussians at the same spatial locations to isolate the effect of Gaussian shape from that of spatial location.

4 Experimental Evaluation

4.1 Experiment Setup

Implementation Details. We implement AGD in PyTorch based on the official 3DGS (Kerbl et al., 2023) implementation used by Poison-splat (Lu et al., 2025). Poisoned training data and victim training settings follow the Poison-splat framework, with the original 3DGS pipeline used as the baseline. All experiments are conducted on a single NVIDIA RTX 2080 Ti GPU. For fair comparison, all methods use the same training images, camera poses, optimization schedule, densification interval, and training iterations.

The default spectral-analysis settings use a 16×16 tile size and a normalized radial frequency range of $[0.3, 0.9]$. STI uses a maximum threshold scaling ratio of 1.5. The logistic modulation uses $(A_T, \delta) = (0.18, 0.05)$ for COLMAP scenes and $(0.3, 0.1)$ for Blender scenes, are fixed hyperparameters selected on a held-out validation set. DGA constrains the weighting factor to $[1.0, 1.5]$. For SR, the thresholds (τ_ν, τ_γ) are set to $(0.78, 0.6)$ for COLMAP scenes and $(1.55, 0.3)$ for Blender scenes. SR is activated from iteration 5,000 onward with $\lambda_s = 1$ and evaluated every 50 iterations using visible Gaussians only.

For Poison-splat experiments, poisoned training sets are generated following its default attack configuration with perturbation strength $\epsilon = 16$, unless otherwise specified. All methods are evaluated under both clean and poisoned settings. Each configuration is independently trained three times, and quantitative results are averaged across the three runs.

Datasets and Metrics. We evaluate AGD on Mip-NeRF360 (MIP) (Barron et al., 2022), NeRF Synthetic (NS) (Mildenhall et al., 2021), and Tanks and Temples (TT) (Knapitsch et al., 2017), covering both synthetic and real-world scenes. Due to computational constraints, MIP and TT images are downsampled during training. Both clean and poisoned settings are evaluated to measure rendering preservation and defense effectiveness, respectively.

PSNR and SSIM measure novel-view rendering quality. To evaluate the computational impact of Poison-splat, we additionally report Maximum Gaussian Number (M), Maximum GPU Memory (MB), and Training Time (min). These metrics jointly measure Gaussian proliferation and its resulting memory and computational overhead.

4.2 Results and Comparisons

Table 1 summarizes the results under Poison-splat attacks. AGD consistently suppresses excessive Gaussian proliferation across all three datasets, resulting in lower peak GPU memory consumption and shorter training time. This demonstrates that regulating the densification process can effectively

Table 1: Comparison between original 3DGS and AGD under Poison-splat attacks. Values in parentheses indicate the reduction achieved by AGD relative to original 3DGS.

Dataset	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
	3DGS	AGD	3DGS	AGD	3DGS	AGD	3DGS	AGD	3DGS	AGD
NS	33.33	33.04	0.798	0.789	0.715	0.561 (**-21.5%**)	6832	3356 (**-50.9%**)	15.86	12.77 (**-19.5%**)
TT	28.59	28.25	0.889	0.879	1.852	1.337 (**-27.8%**)	4912	3798 (**-22.7%**)	18.08	14.02 (**-22.5%**)
MIP	31.52	31.22	0.892	0.882	3.595	2.985 (**-17.0%**)	10112	8747 (**-13.5%**)	34.83	29.68 (**-14.8%**)

Table 2: Compatibility of AGD with existing defenses on NeRF Synthetic. C and P denote clean and poisoned training, respectively.

Defense	AGD	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
		C	P	C	P	C	P	C	P	C	P
Image Smoothing	✗	34.78	38.43	0.986	0.974	0.213	0.268	2088	2306	7.79	9.46
	✓	34.77	38.36	0.986	0.974	0.218	0.263	2046	2137	7.71	9.04
RemedyGS	✗	36.72	40.16	0.959	0.973	0.155	0.157	2081	2080	8.97	8.87
	✓	36.73	40.35	0.953	0.973	0.160	0.184	2106	2198	9.09	9.47
DefenseSplat	✗	32.85	32.20	0.969	0.859	0.125	0.143	1875	1915	7.98	8.41
	✓	32.92	32.09	0.969	0.857	0.140	0.128	1928	1914	8.22	8.66

mitigate the computational overhead induced by the attack. AGD introduces a moderate reduction in rendering quality under poisoned training, with the PSNR difference remaining below 0.35 dB across all datasets. Overall, AGD reduces the computational cost of Poison-splat with only a modest degradation in rendering quality.

To further evaluate the compatibility of AGD with existing defenses, we combine AGD with image smoothing (Lu et al., 2025), RemedyGS (Li et al., 2026), and DefenseSplat (Qiao et al., 2026b). These methods intervene at different stages of the 3DGS pipeline, while AGD regulates gradient-driven densification during optimization. Table 2 reports the results on NeRF Synthetic; additional results on Mip-NeRF360 and Tanks and Temples are provided in the Appendix. The results show that adding AGD generally preserves the reconstruction behavior of the underlying defenses while reducing abnormal Gaussian growth under poisoned training. The magnitude of this effect varies across defense types, indicating that AGD is complementary to, rather than a replacement for, existing defense mechanisms.

To evaluate the generality of AGD, we integrate it with four representative 3DGS methods, including Scaffold-GS (Lu et al., 2024), Mip-Splatting (Yu et al., 2024), GDAGS (Zhou et al., 2026), and Speedy-Splat (Hanson et al., 2025a). These methods target different aspects of 3DGS efficiency and representation, rather than being specifically designed for Poison-splat defense. We therefore examine whether AGD can be incorporated into different 3DGS pipelines to regulate excessive densification under poisoned training. Due to space constraints, we report results on NeRF Synthetic in the main paper and provide additional results in the Appendix.

As shown in Table 3, AGD consistently suppresses Gaussian proliferation under poisoned training across the evaluated methods, and generally reduces training time. GPU memory consumption is also reduced in most cases; the exception is GDAGS, where memory usage increases slightly despite the reduced Gaussian count, likely due to interactions between AGD's gradient reweighting and GDAGS's own density-control method. Although this introduces a trade-off in rendering quality, the results demonstrate that AGD can be integrated into diverse 3DGS pipelines to mitigate abnormal densification.

Table 3: Compatibility of AGD with representative 3DGS methods on NeRF Synthetic. C and P denote clean and poisoned training, respectively.

Method	AGD	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
		C	P	C	P	C	P	C	P	C	P
Scaffold-GS	✗	37.59	34.47	0.983	0.902	0.568	4.070	2254	4574	12.78	33.34
	✓	37.52	33.76	0.983	0.890	0.589	3.379	2219	3880	14.53	27.54
Mip-Splatting	✗	37.24	33.87	0.984	0.841	0.293	0.955	2593	10326	11.45	22.80
	✓	37.30	33.54	0.984	0.833	0.312	0.729	2648	8221	11.98	18.70
GDAGS	✗	35.88	34.42	0.980	0.915	0.210	0.935	2071	5901	6.95	14.96
	✓	35.84	33.70	0.980	0.887	0.208	0.666	2030	6360	7.01	13.23
Speedy-Splat	✗	34.03	31.41	0.967	0.748	0.195	0.407	4829	2789	6.18	8.46
	✓	34.11	30.95	0.967	0.738	0.197	0.311	2603	2733	6.64	8.13

Table 4: Mechanism ablation of AGD on NeRF Synthetic. C and P denote clean and poisoned training, respectively.

STI	DGA	SR	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
			C	P	C	P	C	P	C	P	C	P
✗	✗	✗	37.10	33.33	0.982	0.798	0.290	0.715	2219	6832	8.72	15.86
✓	✗	✗	36.91	32.80	0.982	0.781	0.251	0.429	2102	5116	7.88	11.80
✓	✓	✗	37.11	33.04	0.983	0.789	0.296	0.541	2261	5976	8.70	13.57
✓	✓	✓	37.11	33.04	0.983	0.789	0.298	0.561	2191	3356	8.55	12.77

4.3 Ablation Study

Mechanism Ablation. A cumulative ablation study is conducted on NeRF Synthetic to examine the contributions of STI, DGA, and scale regularization (SR). Starting from the original 3DGS, STI first reduces Gaussian proliferation under poisoned training by adapting the densification threshold to scene-level spectral characteristics, reducing the number of Gaussians from 0.715M to 0.429M. Adding DGA partially recovers rendering quality, improving the poisoned-scene PSNR and SSIM from 32.80 to 33.04 and from 0.781 to 0.789, respectively, at the cost of a moderate increase in the number of Gaussians. Finally, adding SR substantially reduces GPU memory consumption from 5976 MB to 3356 MB while preserving the rendering quality achieved by +STI+DGA, with identical PSNR and SSIM. The complete results are reported in Table 4, while additional mechanism and parameter ablations are provided in the Appendix.

5 Conclusion

Anisotropy-Gated Densification (AGD) was introduced as a defense against Poison-splat attacks on 3D Gaussian Splatting. By incorporating scene-aware and direction-aware spectral cues into the densification process, AGD suppresses excessive Gaussian proliferation and its associated computational overhead while maintaining comparable reconstruction quality. Experiments across Mip-NeRF360, NeRF Synthetic, and Tanks and Temples further demonstrate the effectiveness of this approach under poisoned training and its compatibility with existing defenses and representative 3DGS methods. Despite these improvements, AGD currently relies on spectral statistics obtained before optimization and may introduce a trade-off between densification suppression and the reconstruction of fine structures. Future work will explore adaptive spectral modeling that evolves during training, as well as the robustness and generality of AGD under diverse attack types and strengths.

AI USE STATEMENT

In this work, we did not use generative AI tools for tasks that require disclosure under ICLR's policy, such as research ideation, methodology design, or the analysis of results. Consistent with ICLR's recommended-disclosure category, we used generative AI tools to assist with drafting and editing the manuscript text, summarizing and organizing related work, creating and modifying scientific figures, editing software code, and discussing paper structure. All AI-assisted content was reviewed and verified by the authors, who take full responsibility for the final content of this work.

REFERENCES

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5460–5469. IEEE, 2022.

Quoc-Anh Bui-Huynh, Thanh Duc Ngo, Xue Geng, Kaixin Xu, Wang Zhe, Xulei Yang, and Ngai-Man Cheung. Characterizing detectability in 3dgs poisoning: A stage-wise benchmark. *arXiv preprint arXiv:2606.03499*, 2026.

Yang Chen, Yi Yu, Jiaming He, Yueqi Duan, Zheng Zhu, and Yap-Peng Tan. Spectral defense against resource-targeting attack in 3d gaussian splatting. *arXiv preprint arXiv:2603.12796*, 2026.

Xiaobin Deng, Changyu Diao, Min Li, Ruohan Yu, and Duanqing Xu. Efficient density control for 3d gaussian splatting. *arXiv preprint arXiv:2411.10133*, 2024.

Xiaobin Deng, Changyu Diao, Min Li, Ruohan Yu, and Duanqing Xu. Improving densification in 3d gaussian splatting for high-fidelity rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 223–232, 2026.

Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. *Advances in neural information processing systems*, 37:140138–140158, 2024.

Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of gaussians. In *European conference on computer vision*, pp. 165–181. Springer, 2024.

Glenn Grubert, Florian Barthel, Anna Hilsmann, and Peter Eisert. Improving adaptive density control for 3d gaussian splatting. *arXiv preprint arXiv:2503.14274*, 2025.

Alex Hanson, Allen Tu, Geng Lin, Vasu Singla, Matthias Zwicker, and Tom Goldstein. Speedy-splat: Fast 3d gaussian splatting with sparse pixels and sparse primitives. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21537–21546. IEEE, 2025a.

Alex Hanson, Allen Tu, Vasu Singla, Mayuka Jayawardhana, Matthias Zwicker, and Tom Goldstein. Pup 3d-gs: Principled uncertainty pruning for 3d gaussian splatting. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5949–5958. IEEE, 2025b.

Jiaxin Hong, Sixu Chen, Shuoyang Sun, Hongyao Yu, Hao Fang, Yuqi Tan, Bin Chen, Shuhan Qi, and Jiawei Li. Gausstrap: Stealthy poisoning attacks on 3d gaussian splatting for targeted scene confusion. *arXiv preprint arXiv:2504.20829*, 2025.

Wenkai Huang, Yijia Guo, Gaolei Li, Lei Ma, Hang Zhang, Liwen Hu, Jiazheng Wang, Jianhua Li, and Tiejun Huang. Can protective watermarking safeguard the copyright of 3d gaussian splatting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 5149–5157, 2026.

Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4220–4230. IEEE, 2024.

Moonsoo Jeong, Dongbeen Kim, Minseong Kim, and Sungkil Lee. Dc4gs: Directional consistency-driven adaptive density control for 3d gaussian splatting. *Advances in Neural Information Processing Systems*, 38:28144–28174, 2026.

Bo-Hsu Ke, You-Zhe Xie, Yu-Lun Liu, and Wei-Chen Chiu. Stealthattack: Robust 3d gaussian splatting poisoning via density-guided illusions. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 27400–27411. IEEE, 2025.

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.

Sieun Kim, Kyungjin Lee, and Youngki Lee. Color-cued efficient densification method for 3d gaussian splatting. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 775–783. IEEE, 2024.

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.

Yongjae Lee, Zhaoliang Zhang, and Deliang Fan. Safeguardgs: 3d gaussian primitive pruning while avoiding catastrophic scene destruction. In *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 8479–8489. IEEE, 2026.

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

Yanping Li, Zhening Liu, Zijian Li, Zehong Lin, and Jun Zhang. Remedygs: Defend 3d gaussian splatting against computation cost attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 33227–33236, 2026.

Xiangrui Liu, Xinju Wu, Pingping Zhang, Shiqi Wang, Zhu Li, and Sam Kwong. Compgs: Efficient 3d scene representation via compressed gaussian splatting. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 2936–2944, 2024.

Jiahao Lu, Yifan Zhang, Qiuhong Shen, Xinchao Wang, and Shuicheng Yan. Poison-splat: Computation cost attack on 3d gaussian splatting. In *International Conference on Learning Representations*, volume 2025, pp. 98599–98630, 2025.

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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20654–20664. IEEE, 2024.

Linjie Lyu, Ayush Tewari, Jianchun Chen, Thomas Leimkühler, and Christian Theobalt. Faster 3d gaussian splatting convergence via structure-aware densification. *arXiv preprint arXiv:2604.28016*, 2026.

Yanzhe Lyu, Kai Cheng, Xin Kang, and Xuejin Chen. Resgs: Residual densification of 3d gaussian for efficient detail recovery. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 28093–28102. IEEE, 2025.

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.

Michael Niemeyer, Fabian Manhardt, Marie-Julie Rakotosaona, Michael Oechsle, Daniel Duckworth, Rama Gosula, Keisuke Tateno, John Bates, Dominik Kaeser, and Federico Tombari. Radsplat: Radiance field-informed gaussian splatting for robust real-time rendering with 900+ fps. In *2025 International Conference on 3D Vision (3DV)*, pp. 134–144. IEEE, 2025.

Hyunwoo Park, Gun Ryu, and Wonjun Kim. Dropgaussian: Structural regularization for sparse-view gaussian splatting. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21600–21609. IEEE, 2025.

Yiran Qiao, Yiren Lu, Yunlai Zhou, Rui Yang, Linlin Hou, Yu Yin, and Jing Ma. Advsplat: Adversarial attacks on feed-forward gaussian splatting models. *arXiv preprint arXiv:2603.23686*, 2026a.

Yiran Qiao, Yiren Lu, Yunlai Zhou, Rui Yang, Linlin Hou, Yu Yin, and Jing Ma. Defensesplat: Enhancing the robustness of 3d gaussian splatting via frequency-aware filtering. *arXiv preprint arXiv:2602.19323*, 2026b.

Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. Revising densification in gaussian splatting. In *European Conference on Computer Vision*, pp. 347–362. Springer, 2024.

Evangelos Ververas, Rolandos Alexandros Potamias, Jifei Song, Jiankang Deng, and Stefanos Zafeiriou. Sags: Structure-aware 3d gaussian splatting. In *European conference on computer vision*, pp. 221–238. Springer, 2024.

Peihao Wang, Yuehao Wang, Dilin Wang, Sreyas Mohan, Zhiwen Fan, Lemeng Wu, Ruisi Cai, Yu-Ying Yeh, Zhangyang Wang, Qiang Liu, et al. Steepest descent density control for compact 3d gaussian splatting. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26663–26672. IEEE, 2025.

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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20310–20320. IEEE, 2024.

Ziyang Yan, Yihua Shao, Minwen Liao, Siyu Chen, Nan Wang, Muyuan Lin, Jenq-Neng Hwang, Hao Zhao, Fabio Remondino, and Lei Li. 3dsceneeditor: Controllable 3d scene editing with gaussian splatting. *arXiv preprint arXiv:2412.01583*, 2024.

Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *European conference on computer vision*, pp. 162–179. Springer, 2024a.

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*, pp. 1053–1061, 2024b.

Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19447–19456. IEEE, 2024.

Yixun Zhang, Lizhi Wang, Junjun Zhao, Wending Zhao, Feng Zhou, Yonghao Dang, and Jianqin Yin. 3dgaa: realistic and robust 3d gaussian-based adversarial attack for autonomous driving. *IEEE Transactions on Information Forensics and Security*, 2026.

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*, pp. 326–342. Springer, 2024.

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 *International Conference on Learning Representations*, volume 2026, pp. 21415–21437, 2026.

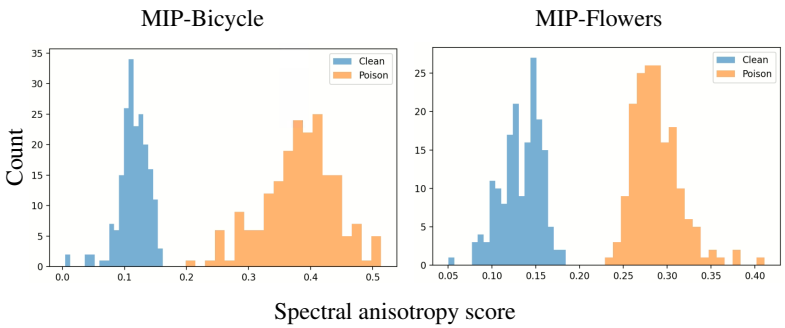


Figure 5: Per-image spectral anisotropy score A for clean and poisoned training images. Poisoned images exhibit a clear shift toward higher anisotropy values, showing a distinct separation from the clean distribution.

A Appendix

This appendix provides additional experimental results and analyses that complement the main paper. We present the spectral anisotropy distribution, per-scene results under clean and poisoned training, compatibility experiments with existing defenses and representative 3DGS methods, and additional ablation studies.

A.1 Spectral Anisotropy Distribution

As described in Section 3.2, the image-level spectral anisotropy score A in Eq. 5 is obtained by aggregating the tile-level anisotropy measurements. Fig. 5 shows the distribution of A across clean and poisoned training images. Poisoned images generally exhibit higher anisotropy values, providing empirical evidence that spectral anisotropy can serve as a signal for adaptive regulation of ADC.

A.2 Additional Experimental Results

We provide additional per-scene results and compatibility evaluations to complement the aggregate results reported in the main paper.

A.2.1 Clean and Poisoned Training

Tables 5 and 6 report the per-scene results of the original 3DGS and AGD under clean and poisoned training conditions. AGD consistently reduces Gaussian proliferation, GPU memory consumption, and training time under poisoned training, while maintaining reconstruction quality close to that of the original 3DGS. The effect is particularly evident on scenes where poisoning causes substantial Gaussian proliferation.

A.2.2 Compatibility with Existing Defenses

The compatibility of AGD with existing defense methods is evaluated on Mip-NeRF360 and Tanks and Temples, with results reported in Tables 7 and 8, respectively.

A.2.3 Compatibility with Representative 3DGS Methods

The compatibility of AGD with representative 3DGS methods is further evaluated on NeRF Synthetic and Tanks and Temples. We integrate AGD with Scaffold-GS (Lu et al., 2024), Mip-Splatting (Yu et al., 2024), GDAGS (Zhou et al., 2026), and Speedy-Splat (Hanson et al., 2025a) and report the corresponding results in Tables 3 and 9. Mip-NeRF360 is excluded from this comparison because some of the evaluated methods exceed the available GPU memory on that dataset.

Table 5: Per-scene results on NeRF Synthetic under clean and poisoned training, while AGD denotes the proposed method under poisoned training. Values report the mean over three runs.

Scene	Condition	PSNR ↑	SSIM ↑	#G (M) ↓	Mem. (MB) ↓	Time (min) ↓
Chair	Clean	39.10	0.994	0.494	2616	10.71
	Poison	34.52	0.820	0.961	6702	17.27
	AGD	34.19	0.810	0.655	3351	13.22
Drums	Clean	29.57	0.977	0.384	2286	9.56
	Poison	30.33	0.777	0.693	5984	14.82
	AGD	30.09	0.768	0.552	3155	12.08
Ficus	Clean	37.90	0.994	0.265	1974	7.72
	Poison	33.64	0.779	0.273	2524	8.88
	AGD	33.38	0.768	0.198	2047	7.63
Hotdog	Clean	40.73	0.991	0.186	2070	8.31
	Poison	33.60	0.790	1.147	10912	24.09
	AGD	33.41	0.786	0.971	4942	18.88
Lego	Clean	40.09	0.992	0.346	2428	9.53
	Poison	34.55	0.832	0.807	8934	16.88
	AGD	34.23	0.824	0.601	3594	13.27
Materials	Clean	35.04	0.987	0.168	1883	7.03
	Poison	32.44	0.767	0.412	3554	10.84
	AGD	32.03	0.761	0.314	2371	9.03
Mic	Clean	40.47	0.997	0.205	2163	7.52
	Poison	33.70	0.755	0.357	5097	11.16
	AGD	33.51	0.748	0.295	2597	9.43
Ship	Clean	33.89	0.926	0.273	2333	9.38
	Poison	33.87	0.861	1.072	10950	22.93
	AGD	33.51	0.850	0.901	4788	18.62
Average	Clean	37.10	0.982	0.290	2219	8.72
	Poison	33.33	0.798	0.715	6832	15.86
	AGD	33.04	0.789	0.561	3356	12.77

A.3 Additional Ablation Studies

Having examined the compatibility of AGD with existing defenses and representative 3DGS pipelines, we further analyze the individual components and parameter sensitivity of the proposed method.

A.3.1 Mechanism Ablation

Tables 10 and 11 extend the mechanism ablation to Mip-NeRF360 and Tanks and Temples. The results are consistent with the observations on NeRF Synthetic: STI provides a substantial reduction in Gaussian proliferation and computational cost, while DGA recovers part of the reconstruction quality. The addition of scale regularization produces relatively small changes in the overall metrics.

A.3.2 Parameter Ablation

Parameter sensitivity is examined on NeRF Synthetic for four key settings: Poison-splat attack strength ϵ, tile size t, scale regularization weight λ_s, and the regularization interval T.

Attack Strength ϵ. AGD is evaluated under Poison-splat perturbation strengths $\epsilon \in \{8, 16, 20\}$. As shown in Table 12, stronger attacks lead to substantially higher Gaussian counts, memory usage, and training time for original 3DGS. AGD reduces these costs across all tested attack strengths with only small differences in reconstruction quality. For $\epsilon = 20$, six scenes are reported because original 3DGS runs out of GPU memory on Hotdog and Mic.

Table 6: Per-scene results on Mip-NeRF360 under clean and poisoned training, while AGD denotes the proposed method under poisoned training. Values report the mean over three runs.

Scene	Condition	PSNR ↑	SSIM ↑	#G (M) ↓	Mem. (MB) ↓	Time (min) ↓
Bicycle	Clean	28.78	0.930	4.061	10119	30.63
	Poison	30.90	0.948	4.066	9394	31.02
	AGD	31.04	0.951	4.416	10296	34.16
Bonsai	Clean	33.68	0.961	1.316	6959	20.74
	Poison	32.56	0.846	2.672	9738	29.26
	AGD	32.02	0.827	1.690	7507	21.57
Counter	Clean	30.56	0.937	1.198	6036	22.41
	Poison	31.04	0.851	2.559	9839	34.92
	AGD	30.67	0.835	1.764	7185	26.35
Flowers	Clean	24.58	0.785	3.440	9335	32.17
	Poison	26.75	0.826	3.753	10090	34.44
	AGD	26.38	0.815	3.196	8883	30.25
Garden	Clean	31.55	0.944	3.729	9491	32.99
	Poison	33.75	0.960	4.196	10756	37.24
	AGD	33.39	0.956	3.471	9430	32.17
Kitchen	Clean	33.73	0.960	1.770	6972	27.07
	Poison	34.08	0.935	3.177	9915	40.04
	AGD	33.62	0.929	2.256	7830	30.79
Room	Clean	34.89	0.956	1.480	6807	21.17
	Poison	33.79	0.863	3.344	10586	37.02
	AGD	33.49	0.853	2.408	8588	28.49
Stump	Clean	33.25	0.955	4.177	9572	30.54
	Poison	34.43	0.961	4.239	9746	30.67
	AGD	34.76	0.964	4.685	10771	34.27
Treehill	Clean	24.83	0.813	3.648	9915	33.07
	Poison	26.34	0.837	4.346	10940	38.88
	AGD	25.57	0.810	2.983	8232	29.03
Average	Clean	30.65	0.916	2.758	8356	27.87
	Poison	31.52	0.892	3.595	10112	34.83
	AGD	31.22	0.882	2.985	8747	29.68

Table 7: Compatibility of AGD with existing defenses on four selected scenes from Mip-NeRF360 (Bonsai, Kitchen, Room, and Treehill). C and P denote clean and poisoned training, respectively.

Defense	AGD	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
		C	P	C	P	C	P	C	P	C	P
Image Smoothing	✗	32.46	34.63	0.937	0.945	1.774	1.405	6912	6171	23.16	20.67
	✓	32.50	34.28	0.939	0.940	1.938	1.133	7420	5706	24.26	18.28
RemedyGS	✗	32.33	33.70	0.933	0.929	1.475	1.466	7346	7196	24.93	25.08
	✓	32.57	33.88	0.936	0.932	1.708	1.689	7852	7709	27.77	27.67
DefenseSplat	✗	32.03	31.51	0.922	0.849	1.664	1.546	7543	7496	27.71	26.70
	✓	32.05	31.17	0.924	0.840	1.718	1.143	7723	6561	28.16	22.43

Tile Size t. Table 13 summarizes the effect of tile size on AGD. Larger tile sizes provide marginally higher reconstruction quality, but generally incur additional memory and computational costs. The differences between $t = 16$ and $t = 24$ are small across the evaluated datasets, while $t = 16$ maintains slightly lower overall cost. We therefore use $t = 16$ as the default setting.

Scale Regularization Weight λ_s. Table 14 evaluates different values of λ_s with the regularization interval fixed to 1. Clean reconstruction quality remains largely stable across the tested values,

Table 8: Compatibility of AGD with existing defenses on four selected scenes from Tanks and Temples (Auditorium, Playground, Train, and Truck). C and P denote clean and poisoned training, respectively.

Defense	AGD	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
		C	P	C	P	C	P	C	P	C	P
Image Smoothing	✗	29.58	31.16	0.954	0.939	0.857	0.900	2613	2759	10.68	11.03
	✓	29.41	30.85	0.951	0.934	0.701	0.639	2276	2111	9.39	9.07
RemedyGS	✗	30.16	32.65	0.947	0.962	0.795	0.488	2604	1425	11.77	8.64
	✓	29.97	32.37	0.945	0.960	0.659	0.410	2360	1326	10.66	8.43
DefenseSplat	✗	30.07	27.49	0.942	0.827	0.797	0.762	2642	2571	12.76	11.88
	✓	29.93	27.36	0.941	0.820	0.664	0.569	2403	2190	11.61	10.67

Table 9: Compatibility of AGD with representative 3DGS methods on four selected scenes from Tanks and Temples (Auditorium, Playground, Train, and Truck). C and P denote clean and poisoned training, respectively.

Method	AGD	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
		C	P	C	P	C	P	C	P	C	P
Scaffold-GS	✗	29.64	29.03	0.946	0.911	3.240	5.518	2964	4593	17.02	25.88
	✓	29.52	28.75	0.945	0.902	2.842	4.272	2880	4054	17.93	24.02
Mip-Splatting	✗	29.81	29.35	0.948	0.918	1.326	2.321	10895	10921	18.23	28.21
	✓	29.62	29.07	0.944	0.913	1.075	1.667	10306	10941	16.40	22.01
GDAGS	✗	28.34	28.79	0.934	0.917	0.824	1.549	2892	4737	8.09	11.48
	✓	28.09	28.45	0.928	0.907	0.621	1.029	2419	3366	6.89	8.85
Speedy-Splat	✗	27.87	27.60	0.904	0.862	0.963	1.622	4612	3732	7.71	10.83
	✓	27.63	27.08	0.896	0.840	0.785	1.160	2833	2803	7.28	8.92

whereas stronger regularization gradually reduces poisoned reconstruction quality while lowering memory usage and training time. We use $\lambda_s = 1$ as the default setting, which provides a balanced trade-off.

Scale Regularization Interval T. Table 15 reports the effect of applying scale regularization at different intervals after iteration 5,000. Applying the regularization too frequently increases training cost, while larger intervals provide similar reconstruction quality with lower overhead. Among the tested settings, $T = 50$ provides a practical balance and is used as the default interval.

A.4 Qualitative Results

The quantitative results are complemented by qualitative comparisons of novel-view renderings. Fig. 6 and Fig. 7 compare the original and +AGD results of different methods under poisoned training across representative scenes.

Table 10: Mechanism ablation of AGD on Mip-NeRF360. C and P denote clean and poisoned training, respectively.

STI	DGA	SR	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
			C	P	C	P	C	P	C	P	C	P
✗	✗	✗	30.65	31.52	0.916	0.892	2.758	3.595	8356	10112	27.87	34.83
✓	✗	✗	30.55	30.91	0.915	0.873	2.642	2.481	8064	8140	26.47	25.91
✓	✓	✗	30.73	31.22	0.918	0.882	2.976	2.980	8692	8699	29.44	29.60
✓	✓	✓	30.72	31.22	0.918	0.882	2.982	2.985	8711	8747	29.43	29.68

Table 11: Mechanism ablation of AGD on Tanks and Temples. C and P denote clean and poisoned training, respectively.

STI	DGA	SR	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
			C	P	C	P	C	P	C	P	C	P
✗	✗	✗	28.57	28.59	0.926	0.889	1.153	1.852	3363	4912	12.69	18.08
✓	✗	✗	28.34	28.07	0.923	0.872	0.863	1.085	2727	3214	10.13	11.86
✓	✓	✗	28.43	28.26	0.924	0.879	0.999	1.336	3075	3783	11.62	14.22
✓	✓	✓	28.44	28.25	0.924	0.879	1.001	1.337	3043	3798	11.47	14.02

Table 12: Effect of Poison-splat attack strength ϵ on NeRF Synthetic.

Attack Strength	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
	3DGS	AGD	3DGS	AGD	3DGS	AGD	3DGS	AGD	3DGS	AGD
$\epsilon = 8$	36.59	36.26	0.914	0.911	0.432	0.333	3574	2368	11.58	9.74
$\epsilon = 16$	33.33	33.04	0.798	0.789	0.715	0.561	6832	3356	15.86	12.77
$\epsilon = 20$	32.44	32.16	0.792	0.779	0.826	0.645	7971	3659	16.90	13.86

Table 13: Effect of tile size on AGD across different datasets. C and P denote clean and poisoned training, respectively.

Dataset	Tile	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
		C	P	C	P	C	P	C	P	C	P
NeRF Synthetic	$t = 8$	37.08	33.07	0.982	0.790	0.292	0.580	2118	3326	8.40	12.97
	$t = 16$	37.11	33.04	0.983	0.789	0.298	0.561	2191	3356	8.55	12.77
	$t = 24$	37.13	33.05	0.983	0.790	0.302	0.561	2153	3219	8.62	12.74
Mip-NeRF360	$t = 8$	30.68	31.14	0.918	0.881	2.916	2.863	8476	8520	28.79	28.85
	$t = 16$	30.72	31.22	0.918	0.882	2.982	2.985	8711	8747	29.43	29.68
	$t = 24$	30.76	31.19	0.919	0.882	3.031	2.993	8862	8810	30.15	29.64
Tanks and Temples	$t = 8$	28.39	28.26	0.923	0.879	0.933	1.346	2860	3767	10.78	13.93
	$t = 16$	28.44	28.25	0.924	0.879	1.001	1.337	3043	3798	11.47	14.02
	$t = 24$	28.50	28.25	0.925	0.879	1.035	1.350	3144	3795	11.64	14.08

Table 14: Effect of the scale regularization weight λ_s on NeRF Synthetic with the regularization interval fixed to 1 iteration. C and P denote clean and poisoned training, respectively.

λ_s	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
	C	P	C	P	C	P	C	P	C	P
0.01	37.10	33.06	0.983	0.791	0.297	0.566	2235	3479	9.72	14.46
0.1	37.09	33.04	0.983	0.790	0.297	0.568	2226	3223	9.57	14.01
1	37.12	32.98	0.983	0.787	0.300	0.564	2217	2976	9.61	13.60
10	37.13	32.90	0.983	0.781	0.302	0.574	2210	2901	9.54	13.55
50	37.09	32.87	0.983	0.779	0.301	0.577	2190	2817	9.59	13.39

Table 15: Effect of the scale regularization interval T on NeRF Synthetic. Scale regularization starts from iteration 5,000. C and P denote clean and poisoned training, respectively.

T	PSNR ↑		SSIM ↑		#G (M) ↓		Mem. (MB) ↓		Time (min) ↓	
	C	P	C	P	C	P	C	P	C	P
1	37.12	32.98	0.983	0.787	0.300	0.564	2217	2976	9.61	13.60
10	37.11	33.04	0.983	0.791	0.298	0.574	2222	3197	8.63	12.79
50	37.11	33.05	0.983	0.790	0.298	0.560	2201	3295	8.61	12.87
100	37.10	33.05	0.983	0.790	0.295	0.542	2212	3418	8.62	13.10



Figure 6: Qualitative comparison on the Mip-NeRF 360 scenes (Bonsai and Treehill). For each scene, the left and right images correspond to the original and +AGD results, respectively.

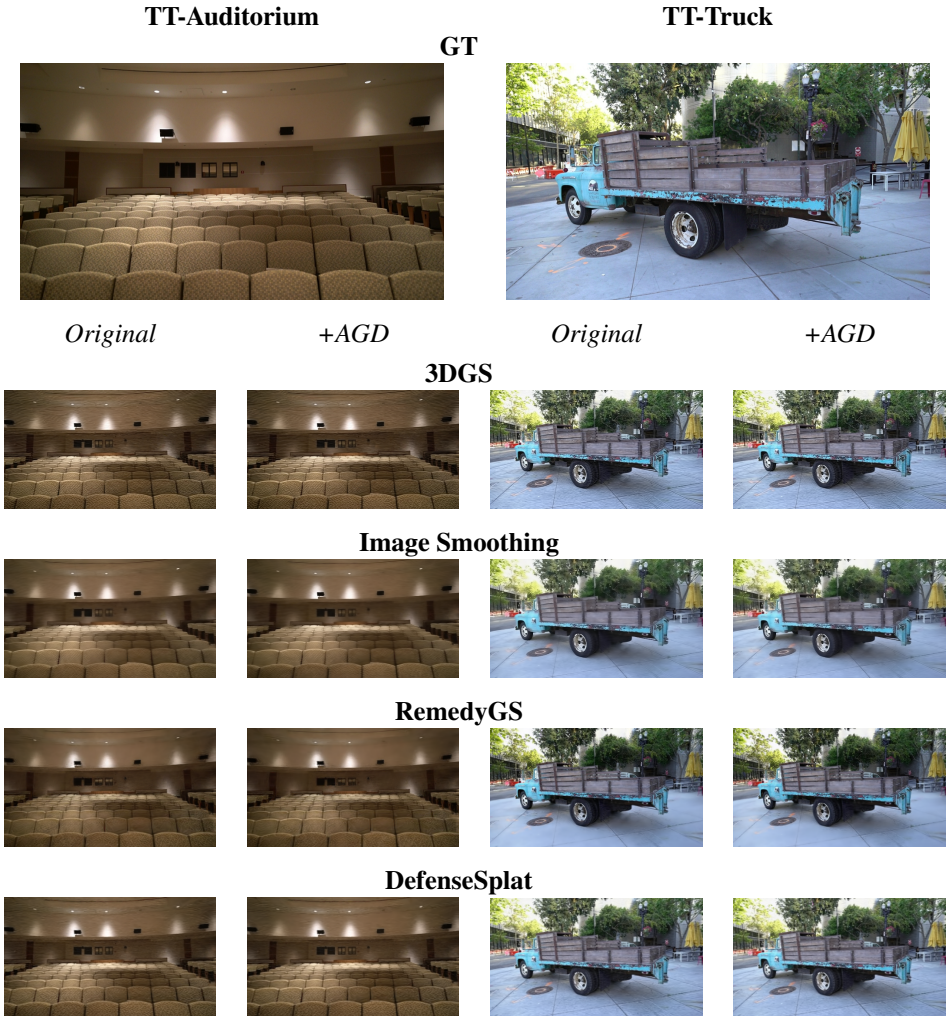


Figure 7: Qualitative comparison on the Tanks and Temples scenes (Auditorium and Truck). For each scene, the left and right images correspond to the original and +AGD results, respectively.