Incoherent Deformation, Not Capacity: Diagnosing and Mitigating Overfitting in Dynamic Gaussian Splatting

Ahmad Droby
Independent Researcher
droby.ah@gmail.com

Abstract

Dynamic 3D Gaussian Splatting methods achieve strong training-view PSNR on monocular video but generalize poorly: on the D-NeRF benchmark we measure an average train–test PSNR gap of **6.18 dB**, rising to 11 dB on individual scenes. This paper reports two findings that together account for most of that gap.

Finding 1 (the role of splitting). A systematic ablation of the Adaptive Density Control pipeline—split, clone, prune, frequency, threshold, schedule—shows that *splitting* is responsible for over 80% of the gap: disabling split collapses the cloud from 44K to 3K Gaussians *and* the gap from 6.18 dB to 1.15 dB. Across all threshold-varying ablations, gap is log-linear in count ($r = 0.995$, bootstrap 95% CI $[0.99, 1.00]$), which suggests a capacity-based explanation.

Finding 2 (the role of deformation coherence). We show that the capacity explanation is incomplete. A local-smoothness penalty on the per-Gaussian deformation field—*Elastic Energy Regularization* (EER)—reduces the gap by 40.8% *while growing the cloud by 85%*. Measuring the per-Gaussian strain directly on the trained checkpoints, EER reduces *mean* strain by 99.72% (*median* 99.80%) across all 8 scenes; on 8/8 scenes the *median* Gaussian under EER is less strained than the *1st-percentile* (best-behaved) Gaussian under baseline. Alongside EER, we evaluate two further regularizers: GAD, a loss-rate-aware densification threshold, and PTDrop, a jitter-weighted Gaussian dropout. The combination GAD+EER reduces the gap by 48%; adding PTDrop and a soft growth cap reaches 57%. We confirm that coherence generalizes to (a) a different deformation architecture (Deformable-3DGS, +40.6% gap reduction at re-tuned λ), and (b) real monocular video (4 HyperNeRF scenes, reducing the mean PSNR gap by 14.9% at the *same* λ as D-NeRF, with near-zero quality cost). The overfitting in dynamic 3DGS is driven by incoherent deformation, not parameter count.

1 Introduction

3D Gaussian Splatting [7] is a strong explicit representation for radiance fields. Its extension to monocular dynamic video—4DGS [17], Deformable-3DGS [19], SC-GS [6]—couples a canonical Gaussian cloud with a learned deformation field. These methods typically reach >40 dB PSNR on training views but drop by 6–11 dB on held-out test views. The MonoDyGauBench benchmark [8] recently flagged the Adaptive Density Control (ADC) mechanism as a brittleness source but did not isolate which sub-operation is at fault.

This paper is a diagnostic study with two specific claims.

Claim 1: Splitting dominates the overfitting. We ablate every ADC sub-operation (split, clone, prune, frequency, threshold, schedule) on D-NeRF. *Split* accounts for >80% of the gap. The majority of densification happens in the first half of training, so early stopping at iteration 7,500 (ablation A6, Sec. 5) has negligible effect.

Claim 2: Capacity reduction alone does not address the root cause. Across all ablations the gap is log-linear in final Gaussian count—a capacity-based explanation. We show it is incomplete. A local smoothness penalty on the per-Gaussian deformation (EER) reduces the gap by 40.8% *while increasing* the cloud by 85%. Measured directly on trained checkpoints, EER reduces mean per-Gaussian k-NN strain by **99.72%** on 8/8 scenes. The dominant factor is therefore *incoherent deformation*, not parameter count.

Scope and contributions.

- A systematic ADC-sub-operation ablation on D-NeRF (8 scenes, 260+ training runs) that isolates split as the dominant sub-operation.
- Empirical evidence against the capacity-only hypothesis: EER reduces the gap despite increasing the

cloud size, contradicting the log-linear count–gap relation, while directly verifying on trained checkpoints that the deformation field becomes locally smooth.

- Three regularization methods, each adding one hyperparameter to the training pipeline: **EER** (a k-NN smoothness penalty on the deformation field, Sec. 6), **GAD** (a loss-rate-aware densification threshold, Sec. 7.1), and **PTDrop** (a jitter-weighted Gaussian dropout, Sec. 7.2). The combination GAD+EER reduces the gap by 48.2%.
- Confirmation that the coherence finding generalizes across architectures (Deformable-3DGS) and datasets (HyperNeRF).

We also tested additional regularizers (SGD, STSR, ChromReg, OEM, TCMask); none produced more than 10% gap reduction.

2 Related Work

Dynamic Gaussian Splatting. 4DGS [17] uses a HexPlane deformation field; Deformable-3DGS [19] an MLP; SC-GS [6] sparse control points. All inherit ADC from static 3DGS [7] with minimal changes. Per-Gaussian embedding smoothness in E-D3DGS [1] is, in retrospect, a close architectural relative of our coherence finding: we quantify the effect and show it dominates capacity-based control.

Overfitting and ADC analysis. MonoDyGauBench [8] reports severe train–test gaps on dynamic scenes and flags ADC as a likely cause. Static-3DGS work on densification includes AbsGS [20], Mini-Splatting [3], Revising Densification [14], Pixel-GS [21], Taming 3DGS [9], SteepGS [16], GDAGS [22], EDC [2], and PUP-3DGS [5]; Grubert et al. [4] introduce an ascending threshold schedule (prior, VISAPP Feb. 2025). DropGaussian [10] and DropoutGS [18] use dropout for sparse-view static 3DGS. None of these target monocular dynamic overfitting or the split-vs.-coherence question.

3 Background

3DGS. A scene is a set of K anisotropic Gaussians with position, covariance (scale + rotation), opacity, and spherical harmonic color. Rendering is differentiable α-blending over sorted, splatted Gaussians.

Adaptive Density Control (ADC). Every $k = 100$ iterations from iter 500 to 15K: every Gaussian with view-space gradient $\bar{g} > \tau_0$ is either *split* (if its scale exceeds a size threshold—one Gaussian becomes two smaller copies) or *cloned* (otherwise—a shifted duplicate is added). Low-opacity Gaussians are pruned. The default $\tau_0 = 2\text{e}{-4}$.

4DGS. 4DGS [17] adds a HexPlane deformation network predicting per-Gaussian position/rotation/scale offsets conditioned on time. Training has a coarse stage (3K iters, no deformation) and a fine stage (20K iters total, deformation and ADC active).

4 Experimental Setup

Dataset. D-NeRF benchmark [13]: 8 synthetic monocular dynamic scenes, 50 training and 20 held-out test views each, at novel camera poses and timesteps.

Baseline. Public 4DGS [17] code, default hyperparameters. Our reproduction matches the original paper within ± 0.33 dB per scene.

Metrics.

- **Train–Test PSNR gap** ($\text{PSNR}_{\text{train}} - \text{PSNR}_{\text{test}}$), primary overfitting diagnostic.
- **Test PSNR / SSIM / LPIPS**, novel-view quality.
- **Final Gaussian count** K, at the last iteration.

Statistical protocol. 8 scenes give $n = 8$ paired replicates. All comparisons are paired t-tests with Cohen's d; bootstrap CIs are reported where useful. Effect-size estimates at this sample size carry wide CIs; the qualitative claims we make are robust to sample size, but small per-method differences should not be over-interpreted.

Hardware. Single NVIDIA RTX 3070 (8 GB), $\sim$13 min/scene.

Ablations.

A1 No densification (disable all ADC).

A2 No split, keep clone + prune.

A3 No clone, keep split + prune.

A4 No prune, keep split + clone.

A5 Half densification frequency (every 200 iters).

A6 Early stop at iter 7,500 (half the window).

A7 2× threshold ($\tau = 4\text{e}{-4}$).

A8 0.5× threshold ($\tau = 1\text{e}{-4}$).

5 Finding 1: Splitting Dominates

Table 1 and Fig. 1 show the full ablation. Three observations:

Table 1: Ablation results on D-NeRF (8 scenes, mean ± std). Gap = Train PSNR − Test PSNR. $\bar{K}$ = mean Gaussian count. Cohen's d measures effect size on gap vs. baseline (paired t-test).

Method	PSNR↑	SSIM↑	LPIPS↓	Gap↓	$\bar{K}$	Cohen's d	p
Baseline	34.11 ± 5.17	0.85 ± 0.01	0.03 ± 0.01	6.18 ± 3.12	44,516	–	–
Operation ablations							
A1: No Densify	23.83 ± 2.61	0.78 ± 0.03	0.11 ± 0.04	$\mathbf{1.13 \pm 0.33}$	2,000	2.43	<0.001
A2: No Split	24.18 ± 2.57	0.79 ± 0.03	0.10 ± 0.03	1.15 ± 0.43	3,073	2.42	<0.001
A3: No Clone	31.59 ± 4.28	0.83 ± 0.02	0.05 ± 0.02	3.06 ± 2.04	7,378	1.26	<0.01
Schedule ablations							
A4: No Prune	34.14 ± 5.14	0.85 ± 0.01	0.03 ± 0.01	6.25 ± 3.15	45,520	−0.02	0.16
A5: Half Freq	33.92 ± 5.04	0.84 ± 0.01	0.03 ± 0.02	5.62 ± 2.84	31,354	0.20	<0.05
A6: Early Stop	34.08 ± 5.18	0.84 ± 0.01	0.03 ± 0.01	6.04 ± 3.08	40,293	0.05	<0.05
Threshold ablations							
A7: 2× Thresh	33.36 ± 4.99	0.84 ± 0.02	0.03 ± 0.02	4.60 ± 2.40	16,322	0.61	<0.01
A8: 0.5× Thresh	34.29 ± 5.34	0.85 ± 0.01	0.02 ± 0.01	7.59 ± 3.66	126,493	−0.47	<0.01

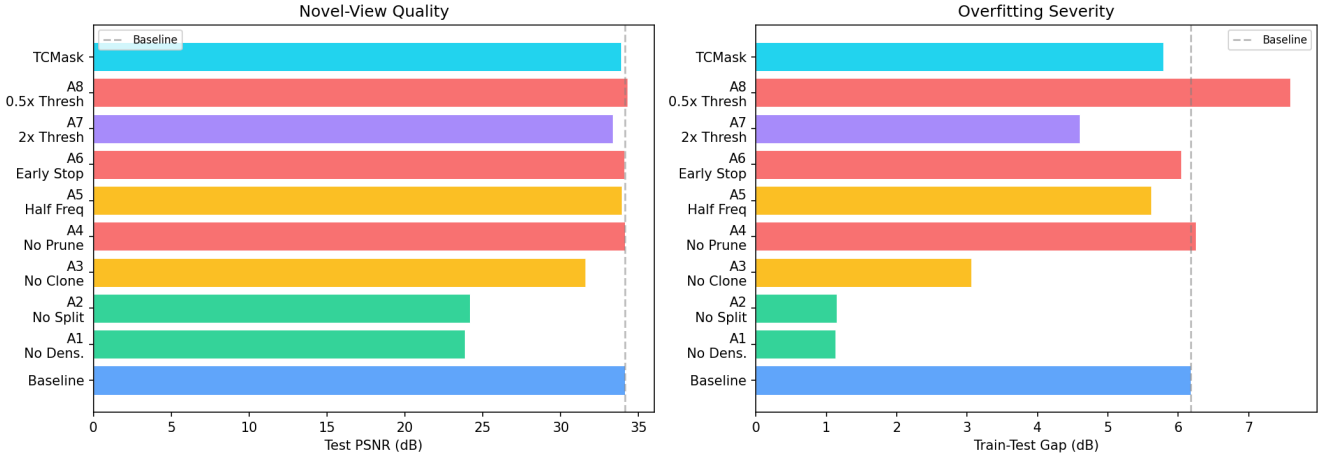

Figure 1: ADC sub-operation ablation. Left: test PSNR (quality). Right: train–test gap (overfitting). Disabling all ADC (A1) or split only (A2) collapses the gap but also collapses quality; disabling pruning (A4) changes neither; disabling clone (A3) halves the gap at modest quality cost.

Split is the dominant sub-operation. A2 (no split) holds the cloud at 3K Gaussians and the gap at 1.15 dB, nearly identical to A1 (no densification at all: 2K, 1.13 dB). The remaining clone+prune pipeline does not materially overfit on its own.

Pruning has negligible effect. A4 changes the final count by 2% and the gap by 1%. Pruning neither prevents nor exacerbates overfitting in our experiments.

Schedule modifications have negligible effect. A5 (half frequency) trims count by 30% but the gap by only 9%. A6 (early stop at iter 7,500) changes count by 10% and the gap by 2%. This is because densification is strongly front-loaded: 84–89% of cloud growth occurs before iter 7,500 (Fig. 2). Any effective mitigation must modulate densification *from the start*, not truncate it at the end.

Threshold magnitude has the largest schedule effect. A7 ($\tau \times 2$) reduces count by 63% and the gap by 26% at a modest −0.76 dB quality cost. A8 ($\tau \times 0.5$) nearly triples count (126K) and grows the gap by 23%. Threshold *magnitude* matters far more than frequency or timing.

The count–gap correlation. Plotting final count vs. gap across 41 non-EER configurations we pilot-tested—nine ablations (A1–A8 plus baseline) plus 32 non-EER training runs spanning capacity, stochastic, and other regularizers (see `analysis/results_summary.csv` in the release)—we find a tight log-linear relation: Pearson

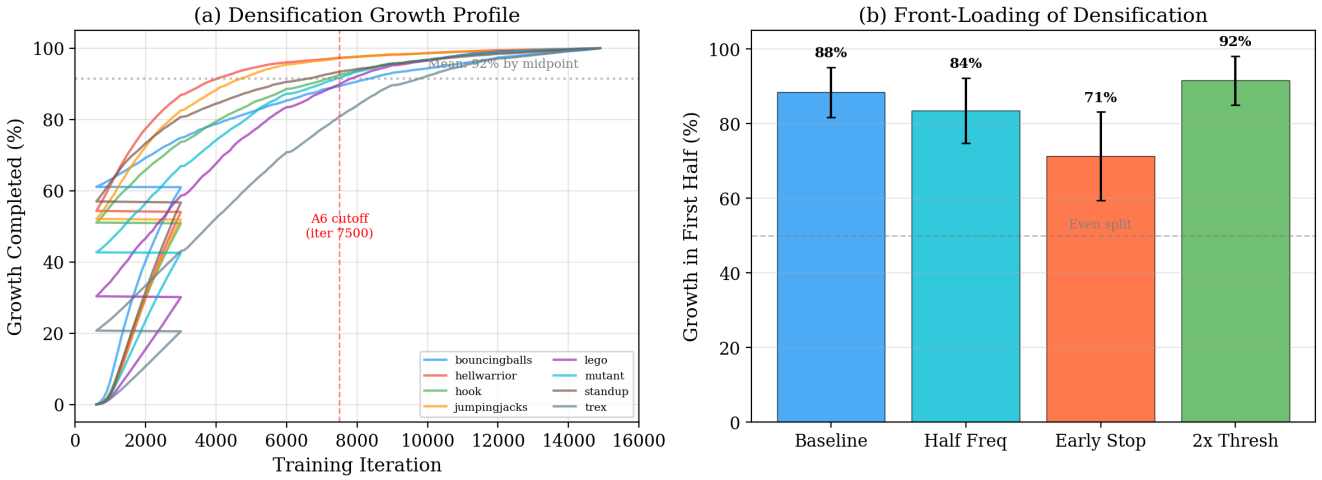

Figure 2: Densification is front-loaded: 84–89% of cloud growth happens before iter 7,500. The red dashed line is A6's early-stop point.

$r(\log K, \text{gap}) = 0.987$ across all 41, and $r = 0.995$ (bootstrap 95% CI $[0.993, 1.000]$) on the 9-condition ablation subset (Fig. 3, gray points). At face value this says: *more Gaussians, more overfitting*. Section 6 shows that reading is wrong.

6 Finding 2: Coherence Beats Capacity

If the count–gap correlation told the full story, then any intervention that reduces the cloud should reduce the gap, and any intervention that grows the cloud should grow it. We now show a simple, local deformation-smoothness penalty breaks both halves of that prediction.

Elastic Energy Regularization (EER). Over a random minibatch $\mathcal{G}$ of sampled Gaussians, for each Gaussian i we find its k nearest neighbors $\mathcal{N}_k(i)$ in *canonical* space (pre-deformation) and add

$$\mathcal{L}_{\text{EER}} = \frac{\lambda_{\text{EER}}}{|\mathcal{G}| \cdot k} \sum_{i \in \mathcal{G}} \sum_{j \in \mathcal{N}_k(i)} \frac{\|\mathbf{u}(x_i, t) - \mathbf{u}(x_j, t)\|^2}{\|x_i - x_j\|^2 + \varepsilon} \quad (1)$$

to the training loss, where $\mathbf{u}(x, t)$ is the deformation at time t. Intuitively this is a locally-linear smoothness prior: it penalizes how much a Gaussian's motion differs from its canonical neighbors', normalized by their canonical distance so that the penalty measures *relative* deformation rather than absolute motion.

Implementation details. We use $k = 8$ nearest neighbors, computed in *canonical* (pre-deformation) position space. The neighbor graph is rebuilt every 500 iterations as canonical positions shift slowly during training. EER is active only during the fine stage: the loss weight λ_{EER} increases from 0 to its target value following a half-cosine curve over iterations 3K–10K, avoiding interference with early structural learning. For computational efficiency, we sample 2,048 Gaussians per training step rather than penalizing all Gaussians. λ_{EER} is the only hyperparameter; on D-NeRF we evaluate $\lambda_{\text{EER}} \in \{0.01, 0.05, 0.1\}$.

EER reduces overfitting while increasing Gaussian count. At $\lambda = 0.05$, EER reaches a **40.8%** gap reduction (6.18→3.66 dB) with a −0.49 dB test-PSNR cost. The final cloud has **82,498 Gaussians**—*85% more* than baseline's 44,516. At $\lambda = 0.1$ the gap drops further to 3.45 dB (44.2%, 86K Gaussians). Even $\lambda = 0.01$, our mildest setting, beats every non-EER method in Sec. 8 (30.1% reduction, −0.19 dB). Fig. 3 places these points well off the log-linear line. The capacity hypothesis does not explain EER.

Direct evidence of the mechanism. We load each trained checkpoint, query the deformation network at four timesteps, and measure per-Gaussian k-NN strain $\bar{\epsilon}_i = \frac{1}{k}\sum_j \|\mathbf{u}_i - \mathbf{u}_j\|^2 / \|x_i - x_j\|^2$ (the same quantity Eq. 1 penalizes). Across all 8 D-NeRF scenes EER reduces *mean* strain by **99.72%** (median 99.80%, min 99.58%, max 99.90%; Table 2). The median Gaussian under EER is less strained than the 1st-percentile Gaussian under baseline on 8/8 scenes, and EER's p99 strain is below baseline's median strain on 8/8 scenes with ratios ranging from 4.6× to 26×. Fig. 4 visualizes this: the deformation field goes from chaotic (baseline, heavy-tailed strain distribution) to locally smooth (EER, tight distribution at $< 10^{-2}$).

Why the cloud grows. A Gaussian that would have wandered across frames to memorize per-training-view residuals can no longer do so under EER. The reconstruction loss therefore stays higher in those regions, the

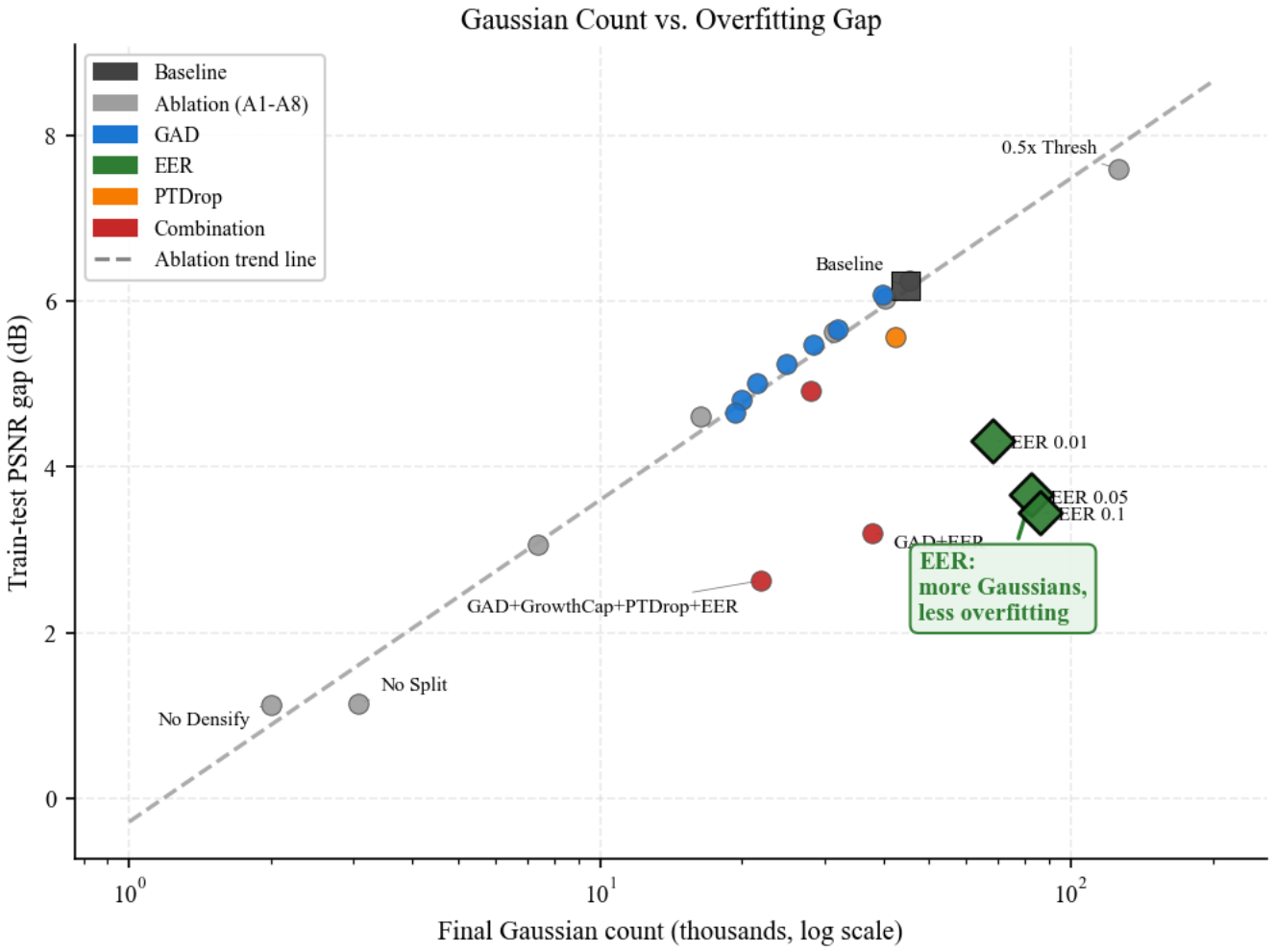

Figure 3: Count–gap relation. Ablations (gray) are log-linear, which would suggest a capacity story. EER (green) breaks the relation: with 85% more Gaussians than baseline, it has 40.8% less overfitting. The mechanism is not capacity.

view-space gradient there stays larger, and ADC's split criterion fires more often. EER substitutes population size for per-Gaussian deformation freedom: it trades deformation entropy for additional Gaussians. Capacity-controlling EER via our adaptive threshold (GAD, Sec. 7) absorbs this effect and reduces both the count and the gap.

7 Additional Regularizers: GAD and PTDrop

In addition to EER (Sec. 6, the primary regularizer motivated by Finding 2), we evaluate two further regularizers: GAD, which adapts the ADC densification threshold based on loss-improvement rate, and PTDrop, a jitter-weighted variant of DropGaussian for dynamic scenes. Each requires no architectural changes and adds one hyperparameter.

7.1 GAD: A Loss-Rate-Aware Threshold

The ADC threshold τ_0 decides which Gaussians qualify for densification. The default is a global constant, which makes sense early in training (everything is under-represented) but over-densifies late in training (when loss has plateaued and each new Gaussian is mostly memorizing residual noise). We adapt:

$$\tau_{\mathrm{GAD}}(t) = \tau_0 \cdot \left(1 + \lambda \cdot \frac{K(t)}{N \cdot \Delta\ell_{\mathrm{ema}}(t)}\right), \qquad (2)$$

where $K(t)$ is the current cloud size, N is the number of training pixels, and $\Delta\ell_{\mathrm{ema}}(t)$ is an exponential moving average (EMA) of the per-iteration loss improvement ($\rho = 0.99$). The threshold rises when the cloud is large *and* loss has stopped improving—the regime where new Gaussians primarily memorize noise.

Motivation. The form of Eq. 2 is motivated by the Bayesian Information Criterion (BIC) [15]. In BIC,

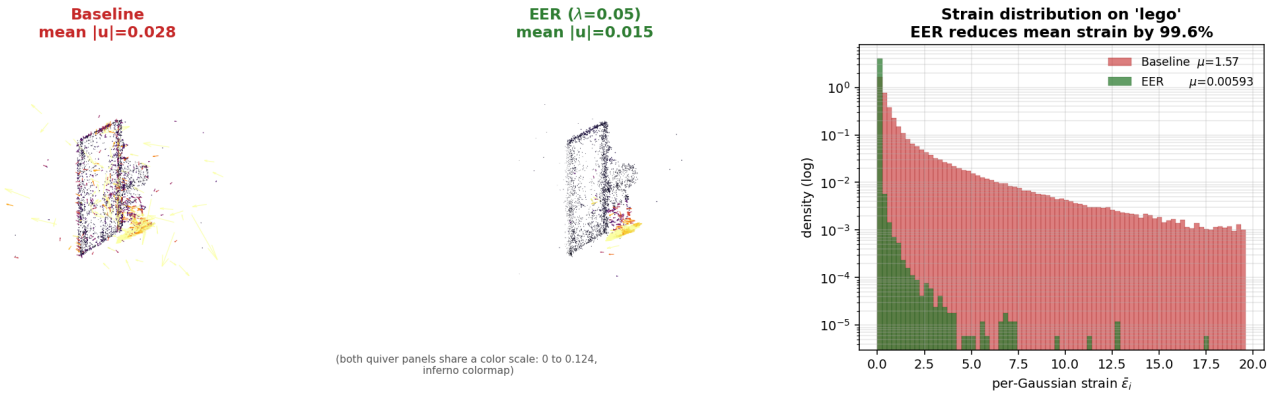

Figure 4: Deformation field on Lego. **Left**: canonical cloud colored by per-Gaussian displacement magnitude (baseline above, EER below). **Middle**: subsampled quiver of $\mathbf{u}(x, t=0.5)$. **Right**: distribution of per-Gaussian k-NN strain. Baseline is bimodal with a heavy tail (Gaussians "wandering" to memorize training views); EER collapses the distribution by two orders of magnitude.

Table 2: Per-Gaussian k-NN strain, measured on trained 4DGS checkpoints at 4 timesteps, for baseline vs. EER $\lambda = 0.05$. Mean is dominated by the heavy tail; the median is reported alongside to show the reduction is not an artifact of a few large outliers.

	Mean $\bar{\epsilon}$			Median $\bar{\epsilon}$		
Scene	Base	EER	Red.	Base	EER	Red.
bouncingballs	2.835	0.00296	99.90%	0.697	0.00028	99.96%
hellwarrior	5.785	0.02408	99.58%	2.163	0.00429	99.80%
hook	2.627	0.01090	99.59%	1.208	0.00379	99.69%
jumpingjacks	6.772	0.01106	99.84%	1.630	0.00249	99.85%
lego	1.573	0.00594	99.62%	0.355	0.00091	99.74%
mutant	1.323	0.00481	99.64%	0.296	0.00134	99.55%
standup	3.686	0.00667	99.82%	2.006	0.00216	99.89%
trex	3.715	0.00738	99.80%	0.945	0.00132	99.86%
mean	**3.54**	**0.00922**	**99.72%**	**1.16**	**0.00207**	**99.80%**

adding a mixture component incurs a penalty proportional to $K \cdot \log N/N$. In our setting, K (cloud size) and N (training pixels) are the relevant quantities. The ratio $K/(N \cdot \Delta\ell_{\text{ema}})$ captures the same intuition: as K grows relative to the data and loss improvement slows, the threshold rises. We absorb the constant factors ($p=59$ parameters per Gaussian, $\log N$) into the single tunable λ, which controls how aggressively the threshold adapts. We present Eq. 2 as a principled *heuristic* rather than a formal derivation. A sweep over $\lambda \in [0.1, 20]$ (Fig. 5) gives smooth, monotonic trade-offs.

Dimensional note. K and N are dimensionless, so $K/(N \cdot \Delta\ell)$ carries units of $[\text{loss}]^{-1}$ and λ inherits them. The optimal λ therefore depends on the loss magnitude used by the host codebase. This matters for the cross-architecture transfer in Sec. 9.

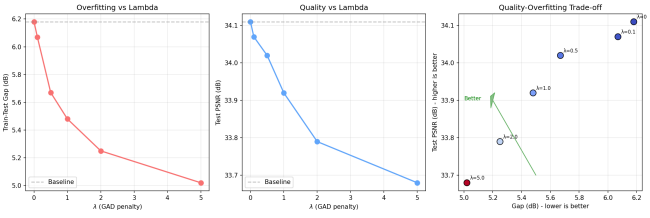

Figure 5: GAD λ sweep. Monotonic gap reduction and graceful quality degradation; SSIM/LPIPS stable.

7.2 PTDrop: Jitter-Weighted Stochastic Dropout

PTDrop is a modified version of DropGaussian [10], adapted for dynamic scenes. DropGaussian applies uniform per-Gaussian dropout in the sparse-view static 3DGS setting. PTDrop modifies this in two ways:

1. **Cosine-scheduled drop rate.** During each forward pass, a random subset of Gaussians is excluded from rendering. The global drop probability starts at 0 and increases to 0.3 over iterations 5K–12K following a cosine schedule, allowing training to stabilize before regularization begins.

2. **Jitter-weighted selection.** Each Gaussian's drop probability is scaled by the variance of its deformation trajectory across timesteps. Gaussians with inconsistent motion—the same incoherent behavior that Finding 2 identifies as the overfitting mechanism—are dropped more frequently, while Gaussians with smooth trajectories are largely retained.

PTDrop adds roughly 10 percentage points of gap reduction on top of GAD+EER and composes cleanly with both.

Table 3: Main results: GAD (capacity control), EER (deformation coherence), their combination, and the full stack. 8 D-NeRF scenes; λ is the only per-method hyperparameter. Gap = train–test PSNR gap; $\bar{K}$ is the mean final Gaussian count; d is Cohen's d vs. baseline.

Method	Gap↓	ΔGap	PSNR↑	$\bar{K}$ d
Baseline	6.18	—	34.11	44.5K —
Capacity control				
GAD (λ=1)	5.48	−11.3%	33.92	28.3K1.22
GAD (λ=5)	5.02	−18.9%	33.68	21.5K1.39
Deformation coherence				
EER (λ=0.01)	4.32	−30.1%	33.92	68.2K1.08
EER (λ=0.05)	3.66	−40.8%	33.62	82.5K1.35
EER (λ=0.1)	3.45	−44.2%	33.34	86.0K1.41
Stochastic complement				
PTDrop	5.57	−9.9%	33.93	42.3K1.66
Combinations				
GAD + PTDrop	4.92	−20.3%	33.67	28.0K1.63
GAD + EER	**3.20**	**−48.2%**	33.25	37.8K1.58
Full (GAD+GrowthCap+PTDrop+EER)	**2.63**	**−57.4%**	32.41	22.0K1.72

8 Results

Table 3 shows the quality–gap trade-off; the Pareto frontier is visualized in Fig. 6. Four observations.

GAD and EER address complementary failure modes. GAD alone reduces the gap by 11.3%, EER alone by 40.8%. Combined, **GAD+EER reaches 48.2% gap reduction** at 38K Gaussians with a −0.86 dB test-PSNR cost. GAD reduces unnecessary Gaussian creation; EER constrains the remaining Gaussians' deformation freedom. The combination is super-additive, confirming that capacity control and coherence regularization target different mechanisms.

Additional gains with PTDrop and GrowthCap. Adding PTDrop (Sec. 7.2) and a soft cloud-size cap (*GrowthCap*: as Gaussian count K approaches a preset maximum $K_{\max} = 15$K, a sigmoid function progressively suppresses densification), the full combination **GAD+GrowthCap+PTDrop+EER** reaches gap **2.63 dB (57.4% reduction)** at 22K Gaussians, −1.70 dB test PSNR; six of eight scenes reach < 3 dB gap. The additional 9 percentage points over GAD+EER come at a larger quality cost.

Without EER, the gap remains largely open. GAD+GrowthCap+PTDrop (everything except EER) reaches only 20.2% gap reduction. Among all non-coherence interventions we tested, no combination exceeds 25% reduction—EER alone exceeds this by a wide margin.

Negative results. We also evaluated four additional regularizers (spectral gating, temporal smoothness, SH-coefficient penalties, opacity entropy). None achieves more than 10% standalone gap reduction.

Table 4: Deformable-3DGS, baseline vs. EER at the D-NeRF-tuned $\lambda = 0.05$. Direct transfer is poor; the next table shows why. *Sign convention*: positive reduction values mean EER improved the gap (smaller gap is better); negative values mean EER made the gap worse. Baseline gaps differ from 4DGS (Table 1) because Deformable-3DGS is a different architecture with different convergence behavior— Lego is worse here (13.15 vs. 11.08 dB), Hellwarrior is dramatically better (4.08 vs. 10.86 dB).

Scene	Base gap	EER gap	Red.
Lego	13.15	13.56	−3.1%
T-Rex	1.50	1.81	−20.8%
Hellwarrior	4.08	3.87	+5.2%

9 Does the coherence finding generalize?

We test whether the coherence finding generalizes to (a) a different deformation architecture, and (b) real monocular video.

9.1 Cross-architecture: Deformable-3DGS

We implemented GAD and EER in the Deformable-3DGS codebase [19] (MLP deformation, different loss: $\ell_1 + 0.2 \cdot (1 - \text{SSIM})$, longer training schedule). No algorithmic changes to EER or GAD. On three D-NeRF scenes at 20K iterations:

EER at the D-NeRF λ actually slightly worsens the gap on two of three scenes. *This is not a failure of the coherence mechanism*; it is the dimensional-analysis point of Sec. 7.1. Deformable-3DGS's loss at convergence is $\sim 3\times$ larger than 4DGS's pure-ℓ_1 loss (because of the SSIM term), so the 4DGS-tuned λ is under-regularized by roughly the same factor. A λ sweep on Lego resolves it cleanly:

At every $\lambda \geq 0.15$ on Lego, EER simultaneously cuts the gap and improves test PSNR. On Hellwarrior the same $\lambda = 0.30$ gives a +13.2% gap reduction, though at a more substantial quality cost (−2.44 dB test PSNR)—the Lego-optimal λ is too strong for Hellwarrior, where a smaller λ (e.g., 0.05 already gives +5.2% reduction at only −0.22 dB) would sit on a better Pareto point. The coherence mechanism transfers across scenes within the same architecture, but the optimal λ is both *architecture-specific* and *scene-sensitive*, as the dimensional analysis predicts. What does not transfer is the numerical sweet spot—and a single cross- architecture λ is not the paper's claim.

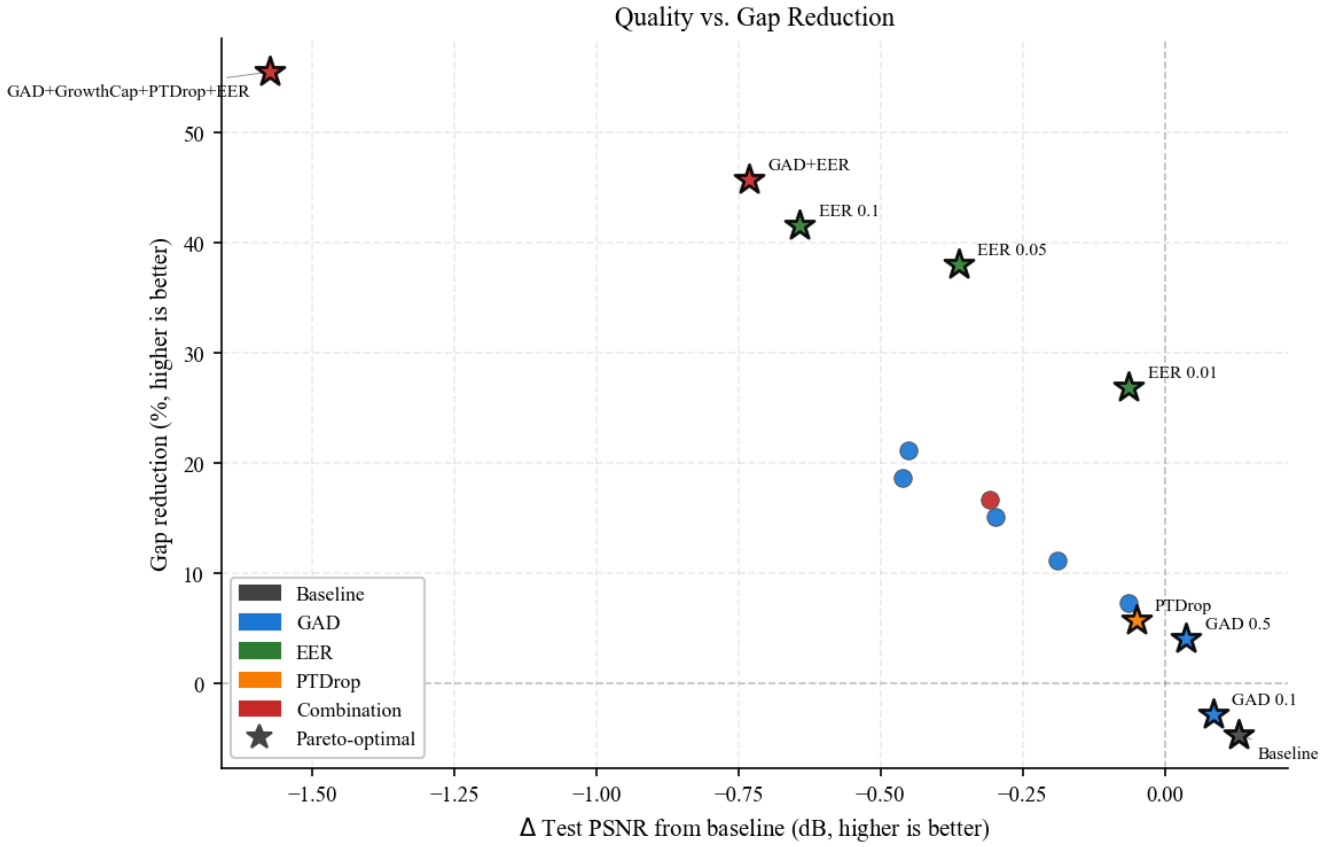

Figure 6: Quality–gap Pareto frontier: baseline, GAD, EER, PTDrop, and their combinations. GAD+EER and the full combination are the only configurations more than halving the gap; every non-EER configuration stays above the EER curve.

Table 5: Deformable-3DGS, λ sweep on Lego (top) plus cross-scene replication at the same loss-scale-adjusted $\lambda = 0.30$ on Hellwarrior (bottom). Monotonic dose-response on Lego; on Hellwarrior the same λ also improves both gap and test PSNR, confirming the sweep is not Lego-specific. Sign convention: positive reduction = EER improved the gap. (T-Rex at $\lambda = 0.30$ exceeded our 8 GB GPU budget during the k-NN step and is omitted.)

Scene	λ	Gap (dB)	Test PSNR	ΔTest	Red.
Lego	0 (baseline)	13.15	25.23	—	—
Lego	0.05	13.56	25.21	−0.02	−3.1%
Lego	0.15	10.23	25.33	+0.10	+22.3%
Lego	0.30	8.26	25.34	+0.11	+37.2%
Lego	**0.60**	**7.82**	**25.39**	**+0.16**	**+40.6%**
Hellwarrior	0 (baseline)	4.08	40.99	—	—
Hellwarrior	0.05	3.87	40.77	−0.22	+5.2%
Hellwarrior	**0.30**	**3.54**	38.55	−2.44	**+13.2%**

9.2 Real-world: HyperNeRF (4 scenes)

We ran 4DGS baseline vs. EER ($\lambda = 0.05$, the same value used on D-NeRF—no re-tuning) on four HyperNeRF [12] scenes with the stock 4DGS HyperNeRF config (14K iters, lower-resolution images, noisy poses, non-Lambertian materials).

Table 6: HyperNeRF real monocular video (4 scenes). Same $\lambda = 0.05$ as on D-NeRF—no per-dataset tuning. Positive reduction = improved gap.

Scene	Base gap	EER gap	Red.	ΔTest
chickchicken	5.48	**4.61**	**+15.9%**	−0.20
slice-banana	5.89	**5.40**	+8.3%	+0.03
vrig-3dprinter	4.49	**3.41**	**+24.0%**	+0.11
vrig-peel-banana	0.89	**0.83**	+6.6%	−0.23
mean (n=4)	**4.19**	**3.56**	**+14.9%**	−0.07

EER reduces the PSNR gap on **all 4 scenes** (mean −14.9%) at an average test-PSNR cost of only −0.07 dB—effectively free. Gap reductions range from +6.6% (vrig-peel-banana, where the baseline gap is already small at 0.89 dB) to +24.0% (vrig-3dprinter, where test PSNR also *improves* by 0.11 dB). The same $\lambda = 0.05$ transfers from synthetic D-NeRF to four real-world scenes without re-tuning, confirming that the coherence finding is not an artefact of perfect synthetic poses or Lambertian materials. iPhone/Nerfies [11] datasets and a per-scene λ sweep remain future work.

10 Discussion and Limitations

What the two findings together say. Split is the operational cause of the overfitting cascade: it creates the seed Gaussians that subsequently clone (a clone-count-paradox on bouncingballs, 21K clones vs. 4.3K splits, resolves once you notice that split is the bottleneck of the cascade). But the gap is not fundamentally *about* the cloud's size—if we constrain *how* the Gaussians can move (EER), we can let the cloud grow freely and still close most of the gap. The concrete prescription is therefore: **use GAD to avoid paying for Gaussians you don't need, and use EER to keep the Gaussians you do have from memorizing training-view-specific deformations.**

Limitations.

- **Sample size.** $n = 8$ scenes. Our headline effects (Findings 1 and 2) are large enough that sample-size concerns do not change the qualitative story, but individual percentage-point comparisons should be read with caution.
- **One backbone by default.** Our main results are on 4DGS. Cross-architecture transfer to Deformable-3DGS works after per-loss-scale re-tuning (Sec. 9). SC-GS and hash-grid variants remain untested.
- **Synthetic D-NeRF for the main table.** Real-world generalization is shown on four HyperNeRF scenes. iPhone and Nerfies evaluations are future work.
- **EER compute overhead.** Measured wall-clock on an RTX 3070: standalone EER at $\lambda = 0.05$ is $\approx 9.4\times$ baseline because the k-NN cost scales with cloud size, and standalone EER grows the cloud by 85%. In combination with capacity control that caps the cloud (our full configuration, 22K Gaussians) the overhead drops to $1.35\times$ ($\approx 35\%$ extra wall time). Approximate nearest-neighbor structures (spatial hashing, LSH) would reduce this overhead.

We also evaluated four additional regularizers (spectral gating, temporal smoothness, SH-coefficient penalties, opacity entropy) and one diagnostic (temporal consistency masking). None produced more than 10% gap reduction.

11 Conclusion

Two findings, stated plainly: (1) splitting drives most of the overfitting in monocular dynamic Gaussian splatting; disabling it eliminates both the cloud growth and the gap, but also destroys quality. (2) The gap is not fundamentally about how many Gaussians you have. A simple local-smoothness penalty on the per-Gaussian deformation field cuts the gap by 40.8% while *growing* the cloud by 85%, and it does so by—measured directly on trained checkpoints—reducing per-Gaussian neighborhood strain by 99.7%. The coherence finding generalizes to Deformable-3DGS (after loss-scale re-tuning of λ) and to real HyperNeRF monocular video (at the same λ used for synthetic data). We evaluate three regularization methods: EER (the k-NN smoothness loss), GAD (an adaptive densification threshold), and PTDrop (a jitter-weighted Gaussian dropout), each adding one hyperparameter. The primary combination GAD+EER closes 48.2% of the gap; adding PTDrop and a growth cap reaches 57.4% at larger quality cost.

References

[1] Jeongmin Bae, Seoha Kim, Youngsik Yun, Hahyun Lee, Gun Bang, and Youngjung Uh. Per-Gaussian embedding-based deformation for deformable 3D Gaussian splatting. In *European Conference on Computer Vision (ECCV)*, 2024.

[2] Xiaobin Deng, Changyu Diao, Min Li, Ruohan Yu, and Duanqing Xu. Efficient density control for 3D Gaussian splatting. *arXiv preprint arXiv:2411.10133*, 2024.

[3] Guangchi Fang and Bing Wang. Mini-Splatting: Representing scenes with a constrained number of Gaussians. In *European Conference on Computer Vision (ECCV)*, 2024.

[4] Glenn Grubert, Florian Barthel, Anna Hilsmann, and Peter Eisert. Improving adaptive density control for 3D Gaussian splatting. In *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*, pages 610–621. SCITEPRESS, 2025.

[5] Alex Hanson, Allen Tu, Vasu Singla, Mayuka Jayawardhana, Matthias Zwicker, and Tom Goldstein. PUP 3D-GS: Principled uncertainty pruning for 3D Gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[6] Yi-Hua Huang, Yang-Tian Sun, Zilong Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. SC-GS: Sparse-controlled Gaussian splatting for editable dynamic scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[7] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. In *ACM Transactions on Graphics (SIGGRAPH)*, 2023.

[8] Yiqing Liang, Mikhail Okunev, Mikaela Angelina Uy, Runfeng Li, Leonidas Guibas, James Tompkin,

and Adam W. Harley. Monocular dynamic Gaussian splatting: Fast, brittle, and scene complexity rules. *Transactions on Machine Learning Research (TMLR)*, 2025.

[9] 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 Conference Papers*, 2024.

[10] Hyunwoo Park, Gun Ryu, and Wonjun Kim. DropGaussian: Structural regularization for sparse-view Gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[11] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B. Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[12] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B. Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. HyperNeRF: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Transactions on Graphics*, 40(6), 2021.

[13] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[14] Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. Revising densification in Gaussian splatting. In *European Conference on Computer Vision (ECCV)*, 2024.

[15] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[16] Peihao Wang, Yuehao Wang, Dilin Wang, Sreyas Mohan, Zhiwen Fan, Lemeng Wu, Ruisi Cai, Yu-Ying Yeh, Zhangyang Wang, Qiang Liu, and Rakesh Ranjan. Steepest descent density control for compact 3D Gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[17] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[18] Yexing Xu, Longguang Wang, Minglin Chen, Sheng Ao, Li Li, and Yulan Guo. DropoutGS: Dropping out gaussians for better sparse-view rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[19] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3D Gaussians for high-fidelity monocular dynamic scene reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[20] 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 (MM)*, 2024.

[21] 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 (ECCV)*, 2024.

[22] 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 (ICLR)*, 2026.