TILESKIPPER: REGION-ADAPTIVE TILE PRUNING FOR 3D GAUSSIAN SPLATTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Tiled 3D Gaussian Splatting rasterizers often use one scene-wide contribution cutoff for tile enumeration, although content differs in its sensitivity to support truncation. TileSkipper selects a static per-Gaussian cutoff policy for a frozen checkpoint. Calibration renders measure candidate pair savings and an isolated-removal distortion proxy that accounts for front transmittance and background color. The method allocates cutoffs across 64 Gaussian groups and accepts policies only after complete renders on disjoint selection views. The exported policy uses one byte per Gaussian, with no parameter updates, additional kernel, or per-frame policy inference. Across 13 scenes from Mip-NeRF 360, Tanks & Temples, and Deep Blending, a fixed-policy AccuTile sweep gives dataset-macro speedups of $\mathbf{1.088}\times$ at standard resolution and $\mathbf{1.238}\times$ at 3840 pixels wide, with $-0.007/-0.023$ dB mean PSNR change. Six integrations with existing opacity-aware bounds yield $1.009\times$–$1.121\times$ compiler-only speedups. For four ports from 3σ rasterizers, we separately attribute the prior exact-bound transition and our incremental gain. Matched-quality ablations show modest gains over scene-global calibration and parity with per-Gaussian control; the standalone comparison with AdaGScale is regime-dependent.

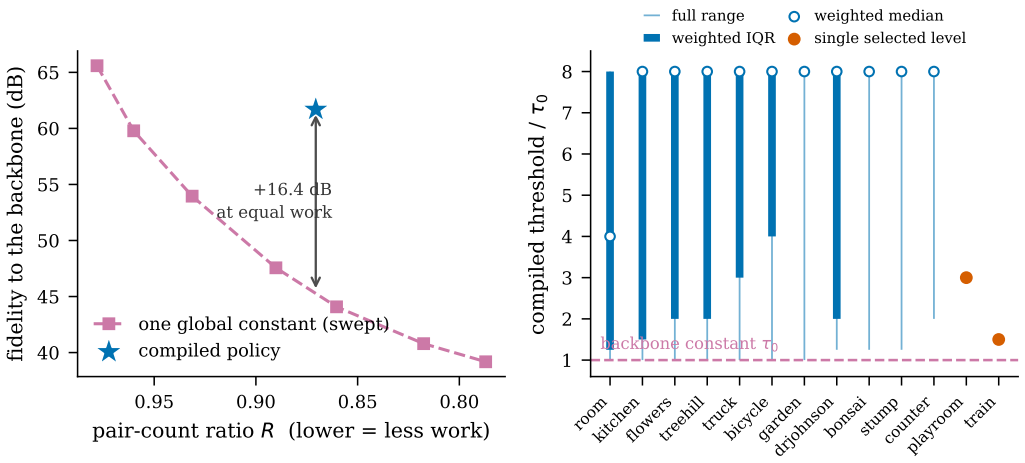
1 INTRODUCTION

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) has become a dominant representation for real-time novel view synthesis. While fast in absolute terms, this pipeline pays a per-frame cost that scales with the number of *(Gaussian, tile) pairs*: a Gaussian with a large projected footprint is replicated across many tiles, and the binning, sorting, and rasterization kernels must process every resulting pair. Reducing this work matters particularly for stereo rendering and for mobile or XR hardware with limited throughput.

A natural way to shrink the work per frame is to test tile intersection against a tighter footprint than the one used for blending. AdR-Gaussian (Wang et al., 2024), Speedy-Splat (Hanson et al., 2025a), and FastGS (Ren et al., 2025) use an opacity level set to replace a conservative projected footprint by a tighter bound. We borrow this level-set construction; the formula itself is not our contribution. Our question begins after the bound is available: how far can the cutoff be raised, differently for each frozen Gaussian, under a measured image-error budget? Raising τ removes low-contribution tails, but their cumulative image error is not guaranteed to be negligible and must be measured. Several threshold-based paths expose only a *single global* τ; whether fixed by the implementation or tuned for a dataset, it cannot allocate an error budget among Gaussians. Flat backgrounds, distant geometry, and nearly opaque material patches tolerate aggressive tile skipping; sharp edges, fine textures, and specular highlights require a conservative threshold. A global rule is forced into a compromise that is too loose on the easy regions and too tight on the hard ones. We make this concrete in Section H: on a single *playroom* view, the largest locally safe threshold spans the full tested range from $\tau^\star = 1/255$ to $16/255$ across regions of the same image.

AdaGScale (Jo et al., 2026) is the closest prior method to this operation. It also performs lossy, viewpoint-adaptive scaling only for tile intersection, uses offline parameter search and a depth lookup table, and retains the original Gaussian during color accumulation. Consequently, training-free operation, a lookup table, and sub-Gaussian support control are not distinctive claims of TileSkipper. The difference is the decision principle: AdaGScale derives a current-view peripheral

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.



(a) the constant traces a curve the policy is not on (flowers, the median scene by this margin)

(b) and the compiled answer is usually spread out (work-weighted quantiles over 64 regions)

Figure 1: **A scene-global cutoff leaves allocation slack.** *(a)* On this diagnostic work–fidelity frontier, the compiled regional policy reaches 16.4 dB higher teacher PSNR than the swept scene-global cutoff at the same pair count; this is the median scene by that margin. The number measures approximation fidelity to the same backbone under equal enumerated work, not a 16.4 dB improvement in ground-truth PSNR. At matched teacher PSNR, Regional is 1.040–1.046× faster than OURS-GLOBAL at standard resolution and 1.041–1.083× faster at 4K (Table 2); it ties the per-Gaussian variant. *(b)* Work-weighted quartiles (thick lines and dots) and full selected ranges (thin lines) over 64 regions. Eleven of thirteen scenes select more than one level; seven span the full τ_0–$8\tau_0$ range, and two collapse to one level. This dispersion shows what a scene-global constant cannot represent; it does not by itself measure end-to-end speedup.

score from local Gaussian contribution and depth, whereas TileSkipper measures candidate deletion residuals over calibration views using the transmittance in front of the Gaussian and the composited color behind it, allocates a discrete error budget across Gaussian groups, and rejects proposed policies by complete renders on disjoint training views. TileSkipper trades AdaGScale's runtime view adaptation for a scene-level, static one-byte policy with no per-frame policy inference. The policy is not indexed by precompiled camera poses: an arbitrary runtime camera is projected and rendered normally using the same level byte, so an interactive viewer requires no camera matching. The checkpoint-matched comparison in Table 3 is dataset-dependent and does not establish broad runtime dominance.

Figure 1 motivates heterogeneous allocation. Calibration renders measure pair savings and a compositing-aware isolated-removal residual for each cutoff. OURS-GLOBAL selects one scene-wide value; OURS-REGIONAL allocates levels across risk-stratified spatial groups. Complete renders on disjoint training views validate every proposal. Regional allocation adds roughly 4–5% over Global at matched native-resolution quality (Table 2). The output is one `uint8` level per Gaussian plus a lookup table, with no checkpoint update or per-frame policy inference. Table 5 evaluates the same exported policies across resolutions, while Section D.5 compares against fixed-threshold alternatives.

The method is deliberately a post-training systems adapter. Its measurements are scene-specific, but the resulting decision is static and portable: policies compiled on AccuTile transfer unchanged into SeeLe (Zhu et al., 2026) and FlashGS (Feng et al., 2025). A measured cost model further separates the pair-dependent work the policy can reduce from fixed work it cannot, and predicts when a complementary geometric pre-culler pays.

Contributions.

- **Compositing-aware support allocation for frozen checkpoints.** For each discrete cutoff, TileSkipper measures exact pair savings and an isolated-removal residual that accounts for front transmittance and the composited color behind the Gaussian, then allocates the image-error budget across risk-stratified, workload-balanced regions. Complete renders on disjoint training views reject proxy failures. The technical distinction from AdaGScale is this measured compositing residual and the resulting error allocation; static one-byte export with no per-frame policy inference is a deployment tradeoff, not the claimed novelty.
- **Evidence across model and rasterizer families.** Six integrations isolate the compiler on paths that already expose an opacity-aware tile bound. Four additional ports begin from released 3σ rasterizers; for these rows we report the deployment-facing total and separately attribute the prior exact-bound transition and the incremental TileSkipper gain. They use the same nine Mip-NeRF 360 (Barron et al., 2022), two Tanks & Temples (Knapitsch et al., 2017), and two Deep Blending (Hedman et al., 2018) scenes. A standalone comparison with AdaGScale fixes the renderer, checkpoint-matched calibration table, selection views, test views, and timing protocol. A Speedy-Splat sweep varies K_{reg}, the number of policy regions, from 1 to 1024 to separate scene calibration from regional allocation.
- **A measured boundary on systems benefit.** A latency model fitted from controlled sweeps predicts the pair-dependent ceiling and the complementary pre-culling regime. The optional pre-culler is image-bit-exact in all 52 measured runs, but is correctly disabled when its planning overhead exceeds the predicted saving.

2 Related Work

We distinguish primitive, tile-pair, and fragment reduction; Appendix A summarizes their control units.

Per-Gaussian thresholds at the fragment level. Fragment Pruning (Ye et al., 2024) learns one truncation threshold per Gaussian via a sigmoid relaxation and controls per-pixel coverage after sorting. TileSkipper selects discrete tile-enumeration cutoffs without updating checkpoint parameters and acts before sorting, removing pairs from both sorting and blending. Section F evaluates TileSkipper on the checkpoint families used by Fragment Pruning; it tests checkpoint compatibility, not composition with learned fragment pruning.

System-level paths. SeeLe (Zhu et al., 2026) combines camera-cluster preprocessing with contribution-aware rasterization. Its tile-intersection test uses the support in Equation (1) with $\tau = 1/255$ fixed at compile time. TileSkipper changes that cutoff policy, while our transfer experiment retains SeeLe's other components and does not claim to reproduce its full mobile deployment pipeline.

Tile-pair reduction. AdR-Gaussian (Wang et al., 2024) derives the opacity level set at the renderer's minimum splatting-opacity cutoff and bounds it with an adaptive radius and an axis-aligned box. We borrow that level-set construction rather than claim the support formula as new. Speedy-Splat (Hanson et al., 2025a) introduces accurate tile pruning, FastGS (Ren et al., 2025) combines it with a compact box, and FlashGS (Feng et al., 2025) and QuadBox (Li et al., 2026) further tighten geometric enumeration. These exact or fixed-threshold bounds are complementary to selecting heterogeneous lossy cutoffs.

View-adaptive tile support. AdaGScale (Jo et al., 2026) is the closest work in action space: it estimates a current-view peripheral contribution score, applies depth-dependent lookup-table calibration, and adjusts the scale used for tile intersection while retaining the original Gaussian for color accumulation. TileSkipper instead measures a compositing residual—including front transmittance and the composited color behind the Gaussian—for discrete candidate cutoffs over calibration views, allocates the resulting error budget across Gaussian groups, and accepts a static policy only after complete-image validation on disjoint training views. AdaGScale offers runtime view adaptation; TileSkipper incurs no per-frame policy inference and directly measures the compiled policy's complete-render error. Table 3 compares the standalone methods inside AdaGScale's released renderer.

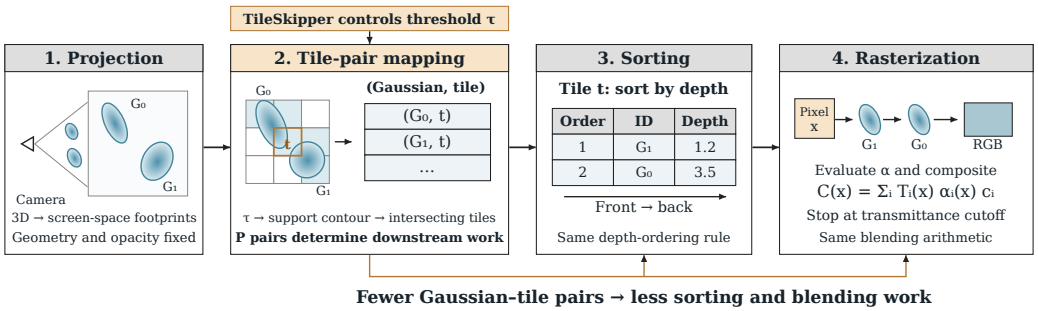


Figure 2: **TileSkipper changes pair enumeration at stage 2.** The blue and orange outlines denote support under the baseline and compiled cutoffs. Tiles excluded by the compiled support generate no Gaussian–tile pair and therefore enter neither sorting nor blending. For tile t, the figure shows the resulting depth order and compositing expression; T_i is transmittance before contribution i.

3 METHOD

TileSkipper takes a converged 3DGS checkpoint and a tile rasterizer whose tile-intersection test uses one scene-wide contribution cutoff τ_0, then replaces that constant with a static per-Gaussian cutoff policy. No checkpoint parameter is retrained. At runtime the rasterizer reads one byte per Gaussian and a lookup table, with no camera-conditioned policy inference. Section 3.1 defines the cutoff and its geometric effect; Section 3.2 measures pair savings and compositing risk, groups Gaussians into workload-balanced regions, selects one cutoff level per region, and validates complete candidate renders. Appendix B provides the empirical cost model, optional pre-culler, and dispatch rule. Figure 3 summarizes the offline measurements, policy selection, and the one-byte runtime application.

Official-Fixed uses the published cutoff τ_0 for every Gaussian. OURS-GLOBAL selects one scene-specific multiplier shared by all Gaussians ($K_{\text{reg}} = 1$), whereas OURS-REGIONAL selects levels for $K_{\text{reg}} = 64$ regions. Here K_{reg} is the number of independently selected policy regions, and m_g denotes Gaussian g's multiplier on τ_0; $m_g = 1$ recovers Official-Fixed.

Rendering pipeline. Figure 2 shows the four stages relevant here. (1) Each visible Gaussian is projected to a screen-space footprint. (2) Every intersecting tile is enumerated, creating one (Gaussian, tile) pair per overlap. (3) Each tile's assigned Gaussians are sorted front-to-back. (4) Pixels evaluate and alpha-blend the sorted Gaussians. Vanilla 3DGS (Kerbl et al., 2023) uses a conservative 3σ box, whereas Speedy-Splat (Hanson et al., 2025a), FastGS (Ren et al., 2025), and SeeLe (Zhu et al., 2026) derive tile support from a fixed contribution cutoff. TileSkipper changes only stage 2; Gaussian parameters, depth sorting, and blending arithmetic remain unchanged. At a fixed checkpoint and resolution, we empirically approximate frame latency as $T \approx F + \beta P$, where P is the Gaussian–tile pair count, F is the fitted pair-independent cost, and β is the fitted latency slope. This model explains why pair reduction need not translate proportionally into runtime improvement; Appendix B.1 gives the full model and validation.

3.1 TILE SUPPORT FROM A SCENE-WIDE CUTOFF

Vanilla 3DGS (Kerbl et al., 2023) enumerates the 3σ box of the projected Gaussian. We borrow the opacity level-set construction used by AdR-Gaussian (Wang et al., 2024): threshold-based rasterizers bound the τ-level support of the splat's contribution. For $0 < \tau < \alpha_g$, the superlevel set $\mathcal{S}_g(\tau) = \{x : \alpha_g(x) \geq \tau\}$ is an elliptical region. With $\alpha_g(x) = \alpha_g \exp(-\frac{1}{2}r^2)$, its boundary $\alpha_g(x) = \tau$ has squared Mahalanobis radius

$$r_g^2(\tau) \;=\; 2\ln\big(\alpha_g/\tau\big). \tag{1}$$

This is the normalized form of AdR-Gaussian's adaptive-radius equation; the level-set formula is prior work, not a TileSkipper contribution. TileSkipper instead contributes the selection of heterogeneous, lossy τ_g values from measured rendering risk. Raising τ shrinks the footprint and removes

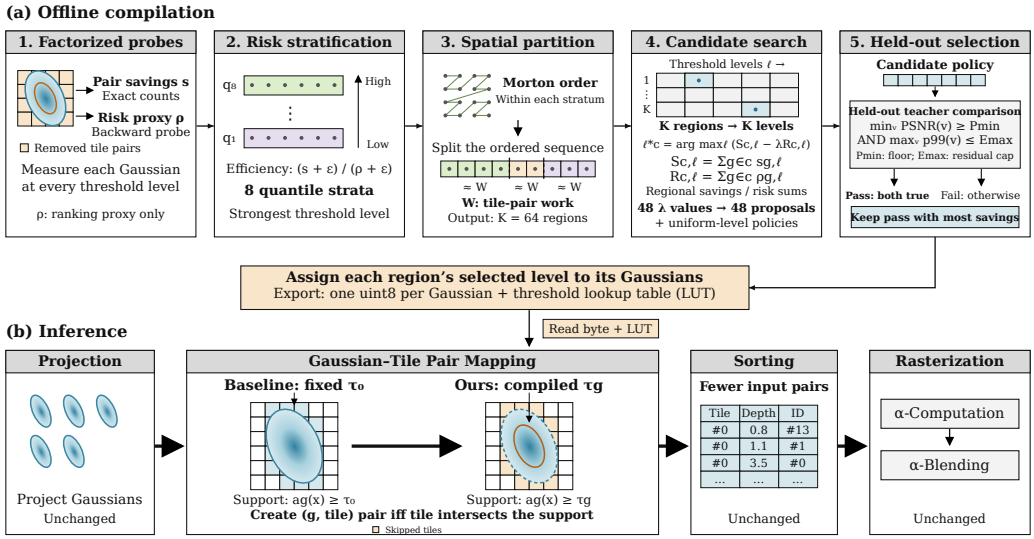


Figure 3: **Offline compilation and runtime application of TileSkipper.** The diagram abbreviates the region-count symbol K_{reg} as K. **(a)** Once per scene, the compiler uses a frozen 3DGS checkpoint and calibration views to measure per-Gaussian pair savings s and isolated-removal risk proxies ρ for each threshold level. The backward probe collects sensitivities without updating any model parameter. Gaussians are assigned to eight efficiency-quantile strata $q_1, \ldots, q_8$ (low to high). Within each stratum, Morton ordering and workload-balanced contiguous partitioning produce $K_{\text{reg}} = 64$ regions in total; Gaussian counts may differ between regions. For region c and level ℓ, $S_{c,\ell} = \sum_{g \in c} s_{g,\ell}$ and $R_{c,\ell} = \sum_{g \in c} \rho_{g,\ell}$ aggregate pair savings and risk. Each penalty λ selects one level for each of the K_{reg} regions. The 48 resulting proposals and uniform-level policies are rendered on disjoint held-out training views. A policy passes only if its minimum per-view teacher PSNR meets $P_{\min}$ and its worst-view p99 absolute residual does not exceed $E_{\max}$; among passing policies, the compiler selects the one with the greatest pair savings. Teacher comparisons use the unmodified backbone; p99 is computed from maximum absolute RGB-channel residuals on a fixed $4\times$ spatial subsample. Test views are never used for selection. **(b)** Each Gaussian inherits its region's selected level, exported as one `uint8` per Gaussian plus a threshold lookup table. At inference, the default threshold is $\tau_g = m_g \tau_0$, where m_g is the selected multiplier and τ_0 is the baseline threshold. Backend-specific handling of opacity caps is described in Section E.1. The contribution field $a_g(x)$ defines the thresholded support $\{x : a_g(x) \geq \tau\}$. A Gaussian–tile pair is created when the tile intersects this support, using τ_0 for the baseline and τ_g for TileSkipper. The dashed blue contour denotes the original support, the orange contour the compiled support, and shaded orange tiles the skipped pairs. Gaussian parameters remain unchanged. The byte and lookup table are read inside tile enumeration, with no per-frame policy inference or additional kernel; projection, sorting, and blending remain unchanged. Footprints, partitions, and table entries are schematic.

pairs. Although every discarded individual contribution is small, their cumulative image error has no such guarantee; this is why our compiler validates complete candidate renders.

The implementations on which we build fix τ to one hand-chosen constant, scene-independent and region-independent:

Rasterizer	Where the constant lives	Value
Speedy-Splat / AccuTile (Hanson et al., 2025a)	tile-intersection test	$\tau_0 = 1/255$
FastGS (Ren et al., 2025)	tile-intersection test	$\tau_0 = 1/255$
SeeLe (Zhu et al., 2026)	`lowest_alpha_coeff`$= \ln 255$ (compile-time)	$\tau_0 = 1/255$

A single constant must protect the most sensitive content represented by its selection views. Section H shows that locally safe thresholds span $16\times$ within one image, establishing that a scene-global value can leave slack in other regions. This diagnostic motivates heterogeneous control but does not

quantify its deployed benefit; the matched-quality comparison in Table 2 measures that benefit directly. TileSkipper replaces the constant with a compiled, static, per-Gaussian value while leaving checkpoint parameters, sorting, and blending unchanged.

3.2 COMPILING A DISCRETE TAIL POLICY

The policy assigns every Gaussian one multiplier from a fixed discrete ladder $\mathcal{M} = \{1, 1.05, 1.1, 1.15, 1.25, 1.5, 2, 3, 4, 6, 8\}$, applied as $\tau_g = m_g \tau_0$ on the default Speedy-Splat path. If the threshold reaches Gaussian opacity, that Gaussian produces no tile pairs; its checkpoint parameters are retained. The FastGS port caps the threshold as detailed in Section E.1. Choosing m_g independently for millions of Gaussians is separable for a fixed Lagrange multiplier but exposes millions of independent decisions. Our default groups them into $K_{\text{reg}} = 64$ decisions as a regularized search granularity, then exports the chosen level for every Gaussian. We compile in five steps.

(1) Factorized probes. For every Gaussian and ladder rung we collect two quantities on calibration views. A global render at rung ℓ records the exact change in enumerated pairs and attributes it to each Gaussian, $s_{g,\ell}$. A custom backward pass classifies pair removal with the same tile predicate and accumulates $\rho_{g,\ell} = \sum_{v,x,c} (\alpha_{g,v}(x)\, \partial C_{v,c}(x)/\partial \alpha_{g,v}(x))^2$ over pixels in pairs removed at rung ℓ. This is an isolated-deletion RGB-energy statistic for removing one Gaussian–tile contribution while holding the others fixed, but it omits cross terms when many contributions are removed jointly. It is therefore a ranking proxy, not an exact prediction of joint image error. Both tables require $O(|\mathcal{M}|V)$ rasterizer passes rather than one render per Gaussian; the held-out stage below validates their joint effect.

(2) Risk stratification. At the most aggressive ladder rung, Gaussians are bucketed into $S = 8$ quantile strata by $\log((s_{g,\ell}+\epsilon)/(\rho_{g,\ell}+\epsilon))$. Gaussians whose tails are cheap to remove can therefore share a decision even when far apart.

(3) Workload-balanced spatial partition. Within each stratum we sort by Morton code and cut into contiguous blocks of *equal measured tile-pair work*, giving K_{reg} regions in total. Balancing on work rather than on volume prevents a few dense regions from consuming most of the decision budget: a uniform axis-aligned grid has a region-work coefficient of variation of 3.1–4.8, which our partition reduces to 0.026–0.060. Region membership is frozen after this step. This balancing is a conditioning choice rather than a claimed source of latency gain; Table 8 isolates its effect.

(4) Constrained search. Aggregating the probes per region gives $S_{c,\ell}$ and $R_{c,\ell}$. For each of 48 penalties λ spaced geometrically over the observed efficiency range, we take the independent optimum

$$\ell_c^\star(\lambda) \;=\; \arg\max_{\ell \in \mathcal{M}} \; \big(S_{c,\ell} - \lambda\, R_{c,\ell}\big), \tag{2}$$

which costs one pass over a $K_{\text{reg}} \times |\mathcal{M}|$ table. The resulting proposals, together with the uniform-ℓ policies that constitute the OURS-GLOBAL candidate set, form the candidate set. Including these uniform candidates lets OURS-REGIONAL fall back to a compiler-selected global cutoff when spatial heterogeneity does not pay.

(5) Held-out selection. Every candidate is rendered on a disjoint set of training views. Feasibility requires the minimum per-view teacher PSNR against the unmodified backbone to exceed a floor and the worst-view absolute p99 residual to be at most a fixed cap. The p99 statistic takes the maximum absolute RGB-channel residual on a fixed $4\times$ spatial subsample. Among feasible candidates we keep the one that removes the most pairs. Test views are never used.

Export. The selected policy is written as one `uint8` level per Gaussian plus a lookup table of at most 256 thresholds — one byte per Gaussian, under 1 KB of table. At render time the rasterizer reads the byte and indexes the table inside the tile-enumeration loop. There is no per-frame policy inference, no additional kernel, and no change to sorting or blending. For the default $K_{\text{reg}} = 64$, compilation averages 9.0 s per scene and takes at most 14.0 s, excluding checkpoint load (Section D.9). It requires only the checkpoint and the training views; the backward probe updates no parameter.

4 EXPERIMENTS

We evaluate ten model/backend integrations on the same thirteen scenes, then isolate policy granularity and compare with AdaGScale. Appendix experiments cover region construction, calibration sensitivity, temporal stability, transfer, and systems costs.

4.1 SETUP

Datasets. We use the standard 3DGS protocol on thirteen scenes: the nine MipNeRF360 scenes (Barron et al., 2022) (*bicycle*, *flowers*, *garden*, *stump*, *treehill*, *bonsai*, *counter*, *kitchen*, *room*), two Tanks & Temples scenes (Knapitsch et al., 2017) (*train*, *truck*), and two Deep Blending scenes (Hedman et al., 2018) (*drjohnson*, *playroom*). We use the official train/test splits and the standard image downsampling used by 3DGS. Standard-resolution test images range from 979×545 to 1559×1039 (0.53–1.62 megapixels); the 4K setting rescales the same views to 3840 pixels wide while preserving aspect ratio.

Backbones and thresholds. Vanilla 3DGS and Mini-Splatting use the same vanilla-compatible CUDA rasterizer; Mini-Splatting changes the surviving Gaussian set but retains the standard PLY representation. Their released 3σ tile box is replaced by the prior exact-$1/255$ bound before our compiled policy is applied. Speedy-Splat is the canonical fixed-threshold AccuTile backbone ($\tau_0 = 1/255$). FastGS combines the same cutoff with its native compact-box rasterizer. We transfer a Speedy-Splat policy without recalibration into the native SeeLe and FlashGS rasterizers. 2DGS uses surfel–ray intersection, while Scaffold-GS generates view-dependent neural Gaussians from anchors; both released rasterizers enumerate a 3σ tile box, so their drop-in ports first supply the exact $1/255$ bound. We separate that prior-work transition from our compiler contribution in Tables 14 and 15. Finally, LightGaussian and Taming-3DGS test pure checkpoints through the same AccuTile backend; the latter retains its native absolute-opacity activation.

Metrics and timing protocol. We report PSNR (Wang and Bovik, 2009), SSIM (Wang et al., 2004), LPIPS (Zhang et al., 2018), the (Gaussian, tile) pair ratio R against the unmodified backbone, and average per-frame render time on one NVIDIA RTX 5000 Ada. We distinguish ground-truth PSNR from *teacher PSNR*. The former compares a rendered image with the dataset image; SSIM and LPIPS use the standard 3DGS evaluation code, with VGG features for LPIPS, and are averaged over the same test views. For an approximate rendering I and the unmodified renderer I_0, both on the $[0, 1]$ RGB scale, teacher PSNR is

$$P_{\text{teacher}} = -10 \log_{10} \left(\underset{x,c}{\text{mean}} (I(x, c) - I_0(x, c))^2 \right).$$

Thus the 60, 55, and 50 dB operating points correspond to teacher-MSE targets of 10^{-6}, $10^{-5.5}$, and 10^{-5}, respectively; 50 dB permits $3.16\times$ the approximation MSE of 55 dB. These targets do not denote PSNR against the dataset images. Every comparison is paired within one backbone and one process; arms are rotated, synchronized per frame, and use identical caching. Speedy-Splat caches its frozen per-Gaussian activations once in every arm. The 2DGS timing includes ray–surfel intersection; the Scaffold-GS timing includes its per-frame MLP Gaussian generation, which our policy cannot reduce. Image writing is disabled. Speedups are geometric means over scenes; cross-row absolute latency is not compared.

Compiler settings. Unless stated otherwise we use the ladder $\mathcal{M} = \{1, 1.05, 1.1, 1.15, 1.25, 1.5, 2, 3, 4, 6, 8\}$, $S = 8$ risk strata, $K_{\text{reg}} = 64$ workload-balanced regions, 48 Lagrangian penalties, and 16 calibration plus 32 disjoint selection views drawn from the training split. Feasibility requires a minimum-view teacher PSNR of 48 dB and a worst-view absolute p99 residual of at most 0.012 on the $[0, 1]$ RGB scale. The teacher is the unmodified base path for each row, except that vanilla 3DGS, Mini-Splatting, 2DGS, and Scaffold-GS use their exact-$1/255$ transition. Test views are never used for calibration or selection. The probe uses backward-mode differentiation but no checkpoint parameter or optimizer state is updated.

Table 1: **Base model/backend versus the same path plus Ours ($K_{\text{reg}} = 64$).** Every row has thirteen scenes and twelve balanced same-process timing cycles. All quality deltas are Ours minus the named base (higher PSNR/SSIM and lower LPIPS are better). R is the mean pair ratio; intervals bootstrap scenes and parentheses count wins. SeeLe and FlashGS transfer a Speedy-Splat policy unchanged into the named native rasterizer. LightGaussian and Taming-3DGS vary the checkpoint on one common AccuTile backend; Taming's native absolute-opacity activation is preserved. The vanilla 3DGS, Mini-Splatting, 2DGS, and Scaffold-GS rows include the prior exact-bound transition, separated in Tables 14 and 15.

Base model/backend	Integration	Scenes	ΔPSNR	ΔSSIM	ΔLPIPS↓	R	+Ours speedup [95% CI] (wins)
Native and native-compatible rasterizer integrations							
vanilla 3DGS (Kerbl et al., 2023)	3DGS CUDA	13	−0.003	−0.000069	−0.000092	0.330	**1.493**× [1.350, 1.656] (13/13)
Mini-Splatting (Fang and Wang, 2024)	3DGS CUDA	13	−0.008	−0.000104	−0.000172	0.435	**1.346**× [1.260, 1.451] (13/13)
Speedy-Splat (Hanson et al., 2025a)	native	13	−0.007	−0.000109	−0.000202	0.825	**1.096**× [1.078, 1.116] (13/13)
FastGS (Ren et al., 2025)	native	13	−0.004	−0.000088	−0.000098	0.979	**1.009**× [1.005, 1.014] (11/13)
SeeLe (Zhu et al., 2026)	native transfer	13	−0.008	−0.000120	−0.000073	0.800	**1.091**× [1.070, 1.115] (13/13)
FlashGS (Feng et al., 2025)	native transfer	13	−0.005	−0.000254	−0.000386	0.821	**1.121**× [1.098, 1.145] (13/13)
2DGS (Huang et al., 2024)	native	13	−0.008	−0.000174	−0.000091	0.641	**1.474**× [1.370, 1.586] (13/13)
Scaffold-GS (Lu et al., 2024)	native	13	−0.008	−0.000132	−0.000115	0.408	**1.320**× [1.218, 1.443] (13/13)
Checkpoint integrations on a common AccuTile backend							
LightGaussian (Fan et al., 2024)	AccuTile	13	−0.006	−0.000123	−0.000196	0.834	**1.095**× [1.072, 1.118] (13/13)
Taming-3DGS (Mallick et al., 2024)	AccuTile	13	−0.001	−0.000038	−0.000031	0.926	**1.051**× [1.032, 1.073] (13/13)

4.2 Main results

Table 1 reports deployment-facing "base versus base + Ours" comparisons, while the accompanying decomposition tables identify which portion is attributable to TileSkipper. On the six paths that already expose an opacity-aware tile bound, the reported gains are compiler-only: $1.096\times$ on Speedy-Splat, $1.009\times$ on FastGS, $1.091\times$ on SeeLe, $1.121\times$ on FlashGS, $1.095\times$ on LightGaussian, and $1.051\times$ on Taming-3DGS. The vanilla 3DGS, Mini-Splatting, 2DGS, and Scaffold-GS rows instead include a prior $3\sigma \to 1/255$ transition. For example, the $1.493\times$ vanilla-3DGS total decomposes into a $1.454\times$ exact-bound transition and a further $1.027\times$ compiler gain; the corresponding Mini-Splatting factors are $1.266\times$ and $1.063\times$. Tables 14 and 15 gives the full accounting. Across all ten integrations, the mean SSIM change is between -3.8×10^{-5} and -2.5×10^{-4}, while LPIPS changes by -3.1×10^{-5} to -3.9×10^{-4} (lower is better).

The 2DGS and Scaffold-GS ports improve by $1.474\times$ and $1.320\times$. The larger 2DGS gain reflects expensive ray–surfel work per removed pair. On Scaffold-GS, the rasterizer itself improves by $2.10\times$, but the per-frame MLP occupies 35–65% of latency and limits the end-to-end result. Because both released paths start from a 3σ box, we expose rather than claim their exact-bound contribution in Table 14.

The gain grows with resolution in the measured regime (Table 5 and Figure 5a). Stage profiles show the duplicate-plus-sort share increasing from 21.4% to 34.2% on *bicycle* and from 31.6% to 41.6% on *counter*; thresholding also changes correlated blending work. Additional ablations, transfer breakdowns, temporal tests, and cost-model diagnostics are reported in the appendix.

4.3 What regional allocation adds

Table 2 compares Global ($K_{\text{reg}} = 1$), Regional ($K_{\text{reg}} = 64$), and per-Gaussian control using the same probes. Each family's measured curve is interpolated linearly in teacher MSE to 60/55/50 dB; these analytical points need not correspond to a single exported policy. Regional is 4.0–4.6% faster than Global at native resolution and up to 8.3% faster at 4K, but ties per-Gaussian control (0.986–$1.007\times$). Thus heterogeneous allocation improves over scene-level calibration, while region sharing has no demonstrated speed or storage advantage over per-Gaussian decisions. Both export one byte per Gaussian; Appendix C gives the full protocol.

4.4 Closest-method comparison

Table 3 compares the two standalone methods under the same training-view quality constraints at native and 4K resolution. These constraints do not imply identical attained errors. At native resolution, the pooled $1.010\times$ speedup has an interval of $[0.990, 1.025]$; this does not establish broad

Table 2: Strict matched-quality ablation of compiler granularity on thirteen scenes. Each variant's own (distortion, latency) Pareto envelope is interpolated at a common teacher-PSNR target against the unmodified backbone. OURS-GLOBAL is our compiler with one decision ($K_{\text{reg}} = 1$), not an external baseline; Regional is the full $K_{\text{reg}} = 64$ method. Brackets are 95% scene-bootstrap intervals. Regional improves on OURS-GLOBAL everywhere and *ties* the per-Gaussian variant. Thus K_{reg} is a decision regularizer and search granularity, not a demonstrated source of speed or storage reduction.

Resolution	Teacher PSNR	Regional / Ours-Global	Per-G / Ours-Global	Regional / per-G	Wins
Standard	60 dB	$\mathbf{1.0397}\times$ [1.0290, 1.0508]	$1.0435\times$	$0.9963\times$	13/13
	55 dB	$\mathbf{1.0457}\times$ [1.0373, 1.0551]	$1.0460\times$	$0.9997\times$	13/13
	50 dB	$\mathbf{1.0401}\times$ [1.0313, 1.0504]	$1.0332\times$	$1.0067\times$	13/13
4K	60 dB	$\mathbf{1.0413}\times$ [1.0208, 1.0612]	$1.0556\times$	$0.9864\times$	10/13
	55 dB	$\mathbf{1.0709}\times$ [1.0500, 1.0923]	$1.0811\times$	$0.9906\times$	13/13
	50 dB	$\mathbf{1.0825}\times$ [1.0632, 1.1029]	$1.0816\times$	$1.0008\times$	13/13

Table 3: **Standalone comparison with AdaGScale at native and 4K resolution.** Both methods use the same Speedy-Splat checkpoints and AdaGScale FlashGS renderer, with policies selected under common training-view quality constraints (details in Section C.2). Deltas are TileSkipper minus AdaGScale; R is the TileSkipper/AdaGScale pair ratio, and speedup is AdaGScale/TileSkipper latency. Brackets are 95% scene-bootstrap intervals; parentheses count speed wins. The two-scene intervals are descriptive.

Dataset	Scenes	ΔPSNR	ΔSSIM	ΔLPIPS↓	R	TileSkipper speedup [95% CI] (wins)
(a) Native resolution						
All scenes	13	+0.0059	+0.000021	−0.000337	0.945	$1.010\times$ [0.990, 1.025] (10/13)
Mip-NeRF 360	9	+0.0056	+0.000066	−0.000349	0.934	$1.017\times$ [1.004, 1.030] (7/9)
Tanks & Temples	2	−0.0001	−0.000032	−0.000161	1.008	$0.960\times$ [0.920, 1.002] (1/2)
Deep Blending	2	+0.0138	−0.000126	−0.000461	0.930	$1.027\times$ [1.024, 1.029] (2/2)
(b) 4K						
All scenes	13	+0.0366	+0.000151	−0.000342	0.929	$1.076\times$ [1.047, 1.102] (12/13)
Mip-NeRF 360	9	+0.0460	+0.000195	−0.000375	0.917	$1.085\times$ [1.061, 1.113] (9/9)
Tanks & Temples	2	+0.0084	+0.000046	−0.000044	0.986	$1.016\times$ [0.952, 1.084] (1/2)
Deep Blending	2	+0.0227	+0.000060	−0.000492	0.924	$1.096\times$ [1.086, 1.106] (2/2)

runtime dominance. At 4K, the pooled speedup is $1.076\times$ $[1.047, 1.102]$, with 12/13 scene wins. Dataset-specific results and all three quality metrics are reported in the same table; Section C.2 details the protocol.

5 CONCLUSION

Across ten model/backend integrations, a scene-wide tile-footprint threshold can be compiled into a heterogeneous static policy rather than hand-tuned. The one-byte-per-Gaussian result performs no training or per-frame policy inference and composes with compact checkpoints and independently optimized rasterizers. Detailed limitations and failure cases appear in Section D.10.

REPRODUCIBILITY STATEMENT

The method section describes the policy construction and selection procedure. The experiments section specifies the datasets, evaluation metrics, compiler settings, and timing protocol. The appendices provide additional details on backend integration, threshold semantics, and experimental protocols to support reproduction of the reported results. We will publicly release the source code and evaluation scripts upon acceptance.

AI USE STATEMENT

Generative AI tools assisted with language polishing. The authors take full responsibility for the entire content of this paper.

REFERENCES

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 (CVPR)*, pages 5470–5479, June 2022.

Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 140138–140158. Curran Associates, Inc., 2024. doi: 10.52202/079017-4447. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/fd881d3b625437354d4421818f81058f-Paper-Conference.pdf.

Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of gaussians. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision – ECCV 2024*, pages 165–181, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-72980-5.

Guofeng Feng, Siyan Chen, Rong Fu, Zimu Liao, Yi Wang, Tao Liu, Boni Hu, Linning Xu, Zhilin Pei, Hengjie Li, Xiuhong Li, Ninghui Sun, Xingcheng Zhang, and Bo Dai. FlashGS: Efficient 3d gaussian splatting for large-scale and high-resolution rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26652–26662, 2025.

Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. Eagles: Efficient accelerated 3d gaussians with lightweight encodings. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision – ECCV 2024*, pages 54–71, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-73036-8. doi: 10.1007/978-3-031-73036-8_4.

Alex Hanson, Allen Tu, Geng Lin, Vasu Singla, Matthias Zwicker, and Tom Goldstein. Speedysplat: Fast 3d gaussian splatting with sparse pixels and sparse primitives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21537–21546, June 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5949–5958, June 2025b.

Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graph.*, 37(6), December 2018. ISSN 0730-0301. doi: 10.1145/3272127.3275084. URL https://doi.org/10.1145/3272127.3275084.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, SIGGRAPH '24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400705250. doi: 10.1145/3641519.3657428. URL https://doi.org/10.1145/3641519.3657428.

Joongho Jo, Hyerin Lim, Hanjun Choi, and Jongsun Park. Adagscale: Viewpoint-adaptive gaussian scaling in 3d gaussian splatting to reduce gaussian-tile pairs, 2026. URL https://arxiv.org/abs/2604.18980.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4), July 2023. ISSN 0730-0301. doi: 10.1145/3592433. URL https://doi.org/10.1145/3592433.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 36(4), July 2017. ISSN 0730-0301. doi: 10.1145/3072959.3073599. URL https://doi.org/10.1145/3072959.3073599.

Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21719–21728, June 2024.

Xinze Li, Bohan Yang, Pengxu Chen, Yiyuan Wang, Hongcheng Luo, Wentao Cheng, and Weifeng Su. Quadbox: Accelerating 3d gaussian splatting with geometry-aware boxes, 2026. URL https://arxiv.org/abs/2605.04844.

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 (CVPR)*, pages 20654–20664, June 2024.

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*, SA '24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400711312. doi: 10.1145/3680528.3687694. URL https://doi.org/10.1145/3680528.3687694.

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)*, pages 134–144, 2025. doi: 10.1109/3DV66043.2025.00018.

Lukas Radl, Michael Steiner, Mathias Parger, Alexander Weinrauch, Bernhard Kerbl, and Markus Steinberger. Stopthepop: Sorted gaussian splatting for view-consistent real-time rendering. *ACM Trans. Graph.*, 43(4), July 2024. ISSN 0730-0301. doi: 10.1145/3658187. URL https://doi.org/10.1145/3658187.

Shiwei Ren, Tianci Wen, Yongchun Fang, and Biao Lu. Fastgs: Training 3d gaussian splatting in 100 seconds, 2025. URL https://arxiv.org/abs/2511.04283.

Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 402–419, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58536-5.

Xinzhe Wang, Ran Yi, and Lizhuang Ma. Adr-gaussian: Accelerating gaussian splatting with adaptive radius. In *SIGGRAPH Asia 2024 Conference Papers*, SA '24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400711312. doi: 10.1145/3680528.3687675. URL https://doi.org/10.1145/3680528.3687675.

Zhou Wang and Alan C. Bovik. Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE Signal Processing Magazine*, 26(1):98–117, 2009. doi: 10.1109/MSP.2008.930649.

Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861.

Zhifan Ye, Chenxi Wan, Chaojian Li, Jihoon Hong, Sixu Li, Leshu Li, Yongan Zhang, and Yingyan (Celine) Lin. 3d gaussian rendering can be sparser: Efficient rendering via learned fragment pruning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 5850–5869. Curran Associates, Inc., 2024. doi: 10.52202/079017-0190. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/0b2de71212384ffcaf80ad9fd1a21fe3-Paper-Conference.pdf.

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

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018. doi: 10.1109/CVPR.2018.00068.

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 (CVPR)*, pages 25979–25989, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Zhu_Seele_A_Unified_Acceleration_Framework_for_Real-Time_Gaussian_Splatting_on_CVPR_2026_paper.html.

A Related-work comparison

Table 4: Positioning by rasterization stage and decision unit. TileSkipper compiles heterogeneous, static per-Gaussian cutoffs without checkpoint updates or a single hand-set scene-wide cutoff; complete held-out renders validate each proposed policy.

Family	Stage	Control unit	Parameter updates
Speedy-Splat (AccuTile) / FastGS (Hanson et al., 2025a; Ren et al., 2025)	Tile-bounding	Whole scene, hand-set	—
AdR-Gaussian / FlashGS / QuadBox (Wang et al., 2024; Feng et al., 2025; Li et al., 2026)	Tile geometry	Per projected Gaussian	—
AdaGScale (Jo et al., 2026)	Tile-bounding	Per view, per Gaussian	—
SeeLe (Zhu et al., 2026)	View prefetch + blending	Camera cluster	fine-tuning
Fragment Pruning (Ye et al., 2024)	Per-pixel rasterization	Per-Gaussian	learned
Gaussian-set pruning (Lee et al., 2024; Girish et al., 2025; Niemeyer et al., 2025; Hanson et al., 2025b)	Gaussian set	Per-Gaussian	varies
View-adaptive scaling (Yu et al., 2024)	Footprint shape	Per-view	—
TileSkipper (ours)	Tile-bounding	**Compiled per-Gaussian**	**none**

B Cost model and optional pre-culling

These diagnostics explain the measured scope of pair reduction and the optional pre-culling extension. The core threshold policy in Section 3.2 can be used without pre-culling.

B.1 A three-term cost model

Following the Gaussian-, pair-, and pixel-level cost decomposition of AdR-Gaussian (Wang et al., 2024), we instantiate a measured affine model for the rasterizers studied here; the decomposition itself is borrowed rather than claimed as a contribution. Let T denote mean end-to-end frame latency, N the number of Gaussians processed per frame, P the number of enumerated Gaussian–tile pairs, and $|\Omega|$ the number of pixels. At a fixed checkpoint and resolution, a global-cutoff sweep changes P while N and $|\Omega|$ remain fixed, so we first fit the effective relation $T = F + \beta P$. Here βP denotes all latency that co-varies with pair count in this controlled sweep, including enumeration, sorting, and correlated downstream rasterization work; it is not a claim that every pair has an identical or context-independent cost. Separate visible-subset and resolution measurements model the remaining variation with N and $|\Omega|$, yielding the empirical affine model

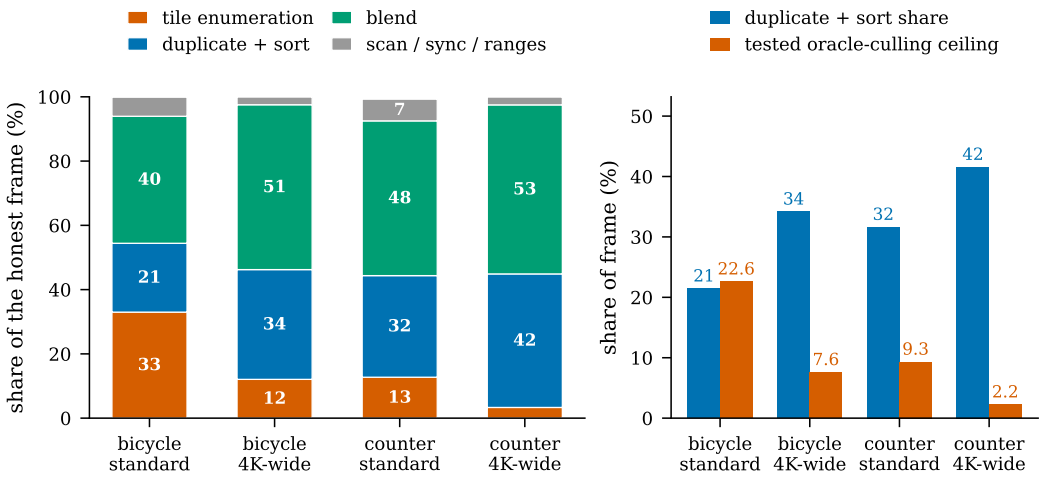
$$T = \underbrace{a\,N}_{\text{per-Gaussian}} + \underbrace{\beta\,P}_{\text{per-pair}} + \underbrace{c\,|\Omega|}_{\text{per-pixel}}. \tag{3}$$

Coefficients are specific to the measured rasterizer, resolution, policy family, and GPU. We use the model for interpolation and dispatch within those controlled sweeps, rather than as a causal decomposition or a hardware-independent law. Figure 4 visualizes the measured stage shares and the terms addressed by thresholding and pre-culling.

Pair count predicts latency within cutoff sweeps. Across eight scene-global cutoff settings in each of nine diagnostic scene–backbone–resolution cells, the linear fit gives $R^2 = 0.9934$–0.9988, where R^2 is the coefficient of determination. Thus pair count explains 99.34%–99.88% of the observed latency variance within these fixed-policy-family sweeps. This supports exactly counted pair removal as a one-dimensional workload surrogate for selecting cutoff policies in the measured regime. It does not establish equal cost per pair or sufficiency of pair count across different policy families, checkpoint densities, or pre-culling configurations.

The per-Gaussian term is invisible to any threshold policy. A cutoff sweep leaves N fixed and therefore cannot reduce per-Gaussian projection and preprocessing. On *bicycle* at native resolution, the preprocessing time has a fitted slope of -0.023 ns per pair and $R^2 = 0.26$. The low R^2 means that pair count explains only 26% of this stage's small timing variation; it is not a hypothesis test that the slope equals zero. The fitted pair-independent preprocessing intercept is 35.1% of baseline frame latency on that checkpoint.

Resolution and density move the measured balance. Increasing resolution changes both P and $|\Omega|$ while leaving N fixed. In the two scenes with stage profiles, the duplicate-plus-sort share grows from 21.4% to 34.2% for *bicycle* and from 31.6% to 41.6% for *counter*; we do not mix the two scenes into one resolution estimate. Increasing checkpoint density can change both N and P, so Gaussian count alone is not a dispatch rule. We instead dispatch from the measured share aN/T.



(a) anatomy of the frame (b) what each mechanism can reach

Figure 4: **Measured headroom under two controlled interventions.** *(a)* Stage shares measured with the rasterizer's own CUDA events. *(b)* The mechanisms have different primary targets, but their runtime effects are not assumed to be disjoint or additive. Thresholding changes tile enumeration, duplicate/sort work, and correlated blending; pre-culling avoids processing rejected Gaussian blocks and may also change downstream work. Within each profiled scene, the duplicate-plus-sort share grows with resolution: $21.4\% \rightarrow 34.2\%$ for *bicycle* and $31.6\% \rightarrow 41.6\%$ for *counter*. The estimated ceiling of the tested perfect, free visibility culler falls from 22.6% to 7.6% and from 9.3% to 2.2%, respectively. Dispatch uses measured net benefit, including planning overhead.

B.2 Conservative region pre-culling

A threshold policy cannot prevent a Gaussian from entering projection and preprocessing. We therefore add a complementary mechanism that skips conservatively rejected Gaussian blocks before projection; its measured image effect is bit-exact rather than error-budgeted.

At export time the model is reordered once by Morton code and cut into fixed-size contiguous blocks, so that spatially adjacent Gaussians are adjacent in memory. For each block we precompute one axis-aligned box that contains the full τ_0-support of every Gaussian it holds, inflated by a safety factor. Each frame, the K'_{reg} block boxes are tested against the camera frustum and the surviving blocks are handed to the rasterizer as contiguous ranges; Gaussians in rejected blocks are never read, so they cost nothing in projection, prefix scan or duplication.

The inflated support boxes are designed to be conservative with respect to the backbone's finite-support predicate. Empirically, the resulting images are bit-identical to the unmodified backbone in all 52 evaluated runs. We therefore call the measured implementation image-bit-exact, while avoiding a claim of numerical exactness for arbitrary cameras or hardware. Planning time is included in every reported pre-culling timing.

B.3 Dispatch: which mechanism, and when

The mechanisms have different primary targets, but their measured savings need not be disjoint or additive. We therefore use Equation (3) as an empirical dispatch model rather than as a causal partition of runtime. Fitting a from one oracle visible-subset comparison provides an estimated ceiling for the tested pre-culling design at that checkpoint, resolution, and hardware. Across thirteen scenes, this estimate matches the realized pre-culling ceiling to a mean absolute error of 0.004.

We enable pre-culling only when its predicted removable time exceeds its measured planning overhead. In our standard-resolution sweep, $aN/T > 20\%$ is a reliable "on" regime and values below 10% are reliably "off"; intermediate cases use the measured inequality rather than a Gaussian-count heuristic. The policy itself remains enabled in both cases. Section 4 reports both arms.

C ADDITIONAL COMPARISON PROTOCOLS AND RESOLUTION RESULTS

C.1 COMPILER GRANULARITY PROTOCOL

Table 9 compares against constants a practitioner might plausibly pick. Table 2 instead compares three granularities of our compiler. OURS-GLOBAL ($K_{\mathrm{reg}} = 1$) automatically selects one scene-level threshold under the same held-out constraints; OURS-REGIONAL ($K_{\mathrm{reg}} = 64$) and the per-Gaussian variant reuse the same measurements but expose more decisions. For each family we build its own $(\text{distortion}, \text{latency})$ Pareto envelope and read off latency at a common teacher-PSNR target, so the arms are compared at equal distortion rather than at equal work. The 60/55/50 dB labels are these teacher-distortion targets. For each method, we interpolate linearly in teacher MSE between adjacent measured policies; the interpolated point summarizes its measured curve and need not coincide with one exported discrete policy.

The full regional compiler improves on OURS-GLOBAL at every target and on $13/13$ scenes at standard dataset resolution, by 4.0–4.6%, rising to 8.3% at 4K. This is the incremental contribution of heterogeneous allocation, not the total compiler gain reported against Official-Fixed in Table 1.

The same table carries an important negative result. Against a per-Gaussian policy compiled from the same probes, region sharing is a tie: 0.986–$1.007\times$ across three targets and two resolutions. The per-Gaussian Lagrangian search is separable and costs $O(N|\mathcal{M}|)$ per penalty, not $|\mathcal{M}|^N$. Regional sharing reduces the number of independently selected decisions to $K_{\mathrm{reg}}|\mathcal{M}|$ and acts as a regularizer, but both implementations currently export one level byte per Gaussian. Thus our evidence supports compilation and heterogeneous control, not a storage or speed advantage of $K_{\mathrm{reg}} = 64$ over per-Gaussian control. $K_{\mathrm{reg}} = 1, 64$, and N in Table 2 expose that policy-granularity spectrum directly.

C.2 ADAGSCALE COMPARISON PROTOCOL

For Table 3, AdaGScale recalibrates its twenty-bin T_{upper} table for each checkpoint and resolution on sixteen training views (per-bin maxima; empty bins set to one), then searches $K_{\mathrm{Ada}} \in [0, 64]$ in eighteen steps. TileSkipper re-evaluates its frozen native-resolution $K_{\mathrm{reg}} = 64$ candidate family. Both select on the same 32 held-out training views, requiring minimum-view teacher PSNR ≥ 48 dB and worst-view p99 ≤ 0.012 relative to the same renderer at cutoff $1/255$. Here p99 is the 99th percentile of per-pixel maximum absolute RGB error on a stride-four grid, maximized over views. These constraints do not enforce equal attained quality. Policies are frozen before evaluation on all test views; timing uses twelve balanced same-process cycles with per-frame CUDA synchronization.

The 4K setting uses 3840-pixel width with the original aspect ratio and consistently scaled camera intrinsics. Reference images are Lanczos-resized from the native images, so this setting measures increased rendering workload, not newly captured image detail.

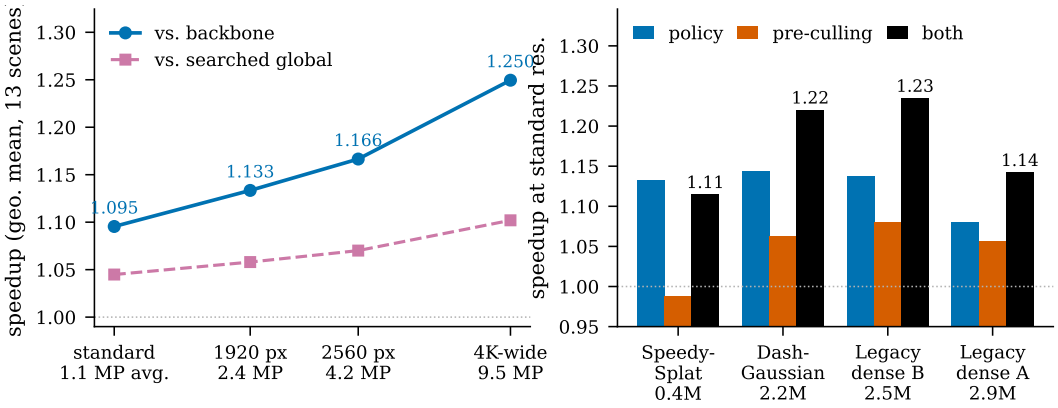
D ADDITIONAL EXPERIMENTS

D.1 FIXED-POLICY RESOLUTION STRESS TEST

Table 5 reports the source of the resolution numbers quoted in the abstract and introduction. It uses one timing protocol consistently at all widths and applies the same held-out-selected policy without recalibration. The standard-resolution geometric mean, $1.0955\times$, agrees with the independent twelve-cycle main-table retiming, $1.096\times$. We report both the equally weighted dataset macro and the thirteen-scene geometric mean so that the abstract's $1.088/1.238\times$ paired comparison is directly traceable. Because the higher-resolution inputs are resampled from the same test views, this experiment establishes scaling with raster workload rather than performance on newly captured 4K detail.

D.2 REGION-COUNT ABLATION ON SPEEDY-SPLAT

Table 6 sweeps K_{reg} from 1 to 1024 on the same thirteen Speedy-Splat scenes, using the existing construction rule with $S_{\mathrm{eff}} = \min(8, K_{\mathrm{reg}})$ risk strata. The $K_{\mathrm{reg}} = 1$ compiler already gives $1.048\times$ over the official fixed cutoff. The added $K_{\mathrm{reg}} = 2$ and 4 policies yield $1.0165\times$ and $1.0351\times$ over



(a) the policy scales with resolution (b) pre-culling needs density

Figure 5: **Each mechanism has a regime, and the cost model names it.** *(a)* The compiled policy strengthens with resolution; both P and $|\Omega|$ change in this sweep. The separate stage profiles in Figure 4 show the within-scene changes and do not conflate the *bicycle* and *counter* endpoints. *(b)* Pre-culling strengthens as the measured aN/T share grows. The dense bars are stress-test checkpoints; the two legacy packaged checkpoints are not used as evidence for any named model family. On Speedy-Splat's 0.4M checkpoint the frustum test costs more than it saves and "both" is *worse* than the policy alone, which is the case the dispatch rule catches.

Table 5: **Fixed-policy resolution stress test on AccuTile, all thirteen scenes.** The same deployable $K_{\text{reg}} = 64$ policies, selected using held-out training views at standard resolution, are applied unchanged at every width; test views are used only for evaluation. Higher-resolution rows bilinearly resample the same source views and therefore test workload scaling rather than natively captured high-resolution content. Frozen per-Gaussian attributes are cached once in both arms. Timing rotates AccuTile, OURS-GLOBAL, and OURS-REGIONAL in one process with three warmup and seven measured repetitions per view. Speedup is AccuTile/Regional latency; "dataset macro" equally weights the three dataset-level arithmetic means, while "scene geom." is the geometric mean over all thirteen scenes. R and all quality deltas are arithmetic scene means over full official test sets; deltas are Regional minus AccuTile (higher PSNR/SSIM and lower LPIPS are better).

Width	MP/frame	R	ΔPSNR	ΔSSIM	ΔLPIPS↓	Speedup (dataset macro / scene geom.)
Standard	1.15	0.8248	−0.0074	−0.000109	−0.000202	1.0879 × /1.0955×
1920 px	2.38	0.8044	−0.0141	−0.000416	−0.000145	1.1283 × /1.1334×
2560 px	4.23	0.7908	−0.0166	−0.000532	−0.000167	1.1615 × /1.1665×
3840 px	9.51	0.7750	−0.0231	−0.000742	−0.000221	1.2384 × /1.2495×

their same-process $K_{\text{reg}} = 1$ controls; $K_{\text{reg}} = 8$ yields 1.0412×. A small number of groups therefore captures most of the observed benefit. For $K_{\text{reg}} = 64$, 128, and 256, the corresponding ratios are 1.0472×, 1.0468×, and 1.0451×. The added $K_{\text{reg}} = 512$ and 1024 rows reach 1.0488× and 1.0519×, with mean pair ratios of 0.8158 and 0.8164 and mean compilation times of 11.2 and 15.0 seconds, compared with 9.0 seconds at $K_{\text{reg}} = 64$. Each ratio uses its own same-process global control; differences between separately timed regional configurations are not direct paired comparisons. These are common-constraint results rather than exact quality matching. We retain the globally fixed $K_{\text{reg}} = 64$ rather than selecting the best K_{reg} per scene. This independent sweep's $K_{\text{reg}} = 64$ speedup (1.095×) agrees with the independently repeated main operating-point retiming (1.096×).

D.3 REGION-COUNT ABLATION ON 2DGS

To test the region-count trend on a different representation, we repeat the ablation on frozen 2DGS checkpoints for the same thirteen scenes, with $K_{\text{reg}} \in \{1, 2, 4, 8, 64, 256, 512, 1024\}$ fixed before test evaluation. Table 7 uses the exact $1/255$ opacity-bound path as its baseline, isolating policy

Table 6: **Region-count ablation on Speedy-Splat, all thirteen scenes.** $K_{\text{reg}} = 1$ is our compiler restricted to one scene-level decision; larger K_{reg} increases the number of policy groups, with spatial workload balancing within each risk stratum. Every row uses the same ladder, calibration/selection views, and quality constraints; the existing construction uses $S_{\text{eff}} = \min(8, K_{\text{reg}})$ risk strata. Latency uses the same honest-cache, rotating three-arm protocol as Table 1; brackets are 95% scene-bootstrap intervals. We retain the globally fixed $K_{\text{reg}} = 64$; the added $K_{\text{reg}} = 2, 4, 512, 1024$ rows use the same three-arm evaluation protocol.

K_{reg}	mean ΔPSNR	R	vs. Official-Fixed	vs. $K_{\text{reg}} = 1$ [95% CI]	compile (s)
1	-0.0041	0.9192	$1.0484\times$	$1.0000\times$	2.8
2	-0.0047	0.8868	$1.0641\times$	$1.0165\times$ $[1.0105, 1.0223]$	3.8
4	-0.0051	0.8479	$1.0843\times$	$1.0351\times$ $[1.0276, 1.0424]$	4.9
8	-0.0067	0.8329	$1.0894\times$	$1.0412\times$ $[1.0298, 1.0525]$	7.0
16	-0.0054	0.8306	$1.0921\times$	$1.0422\times$ $[1.0289, 1.0551]$	7.7
32	-0.0064	0.8273	$1.0949\times$	$1.0436\times$ $[1.0316, 1.0555]$	8.7
64	-0.0080	0.8227	$1.0949\times$	$1.0472\times$ $[1.0327, 1.0604]$	9.0
128	-0.0082	0.8207	$1.0973\times$	$1.0468\times$ $[1.0328, 1.0605]$	9.4
256	-0.0083	0.8201	$1.0957\times$	$1.0451\times$ $[1.0316, 1.0583]$	9.9
512	-0.0093	0.8158	$1.0981\times$	$1.0488\times$ $[1.0345, 1.0632]$	11.2
1024	-0.0078	0.8164	$1.0996\times$	$1.0519\times$ $[1.0380, 1.0667]$	15.0

selection from the earlier geometric-bound improvement. Under the same selection-view quality constraints, speedup rises from $1.0838\times$ at $K_{\text{reg}} = 1$ to $1.1200\times$ at $K_{\text{reg}} = 8$, followed by $1.1216\times$ and $1.1228\times$ at $K_{\text{reg}} = 64$ and 256. The direct paired 256-versus-64 ratio is $1.0011\times$ with a scene-bootstrap interval of $[0.9988, 1.0037]$, while mean compilation time rises from 53.5 to 57.9 seconds. The added $K_{\text{reg}} = 2$ and 4 policies give $1.0144\times$ and $1.0287\times$ over their same-batch $K_{\text{reg}} = 1$ controls. At the other end, $K_{\text{reg}} = 512$ and 1024 give direct paired speedups of $0.9999\times$ $[0.9956, 1.0043]$ and $1.0002\times$ $[0.9938, 1.0066]$ over the same-batch $K_{\text{reg}} = 64$ control, with mean compilation times of 60.5 and 62.6 seconds. Thus increasing the region count beyond 64 has not established an additional runtime benefit in this experiment; this does not establish $K_{\text{reg}} = 64$ as universally optimal. For small K_{reg}, the existing construction uses $\min(8, K_{\text{reg}})$ risk strata, so the 2- and 4-group policies also use fewer strata. The policies attain different distortions under the common constraints, so we report PSNR, SSIM, and LPIPS changes rather than claim exact quality matching.

D.4 Region-construction ablation

The K_{reg} sweep establishes the value of heterogeneous control; we next hold $K_{\text{reg}} = 64$ fixed and change only how Gaussians are assigned to decisions. Random work-balanced groups and a robust $4 \times 4 \times 4$ uniform spatial grid test whether the gain comes merely from adding 64 free parameters. To avoid confounding construction with a different selected distortion, every family is interpolated to the same 55 dB teacher PSNR. Table 8 separates risk stratification, Morton spatial locality, and pair-workload balancing. Random and Uniform-grid retain $88.3\%/88.2\%$ of pairs and reach only $1.065\times$ /$1.066\times$; the full risk-aware construction retains 80.8% and reaches $1.103\times$. Full is faster than Random by $1.036\times$ $[1.029, 1.046]$ and than Uniform-grid by $1.035\times$ $[1.029, 1.041]$, winning all thirteen scenes in both comparisons. Spatial, work-balanced regions without risk stratification lie between them at $1.082\times$, and Full remains $1.020\times$ faster $[1.015, 1.024]$ (13/13). Thus the Pareto gain comes from grouping spatially coherent Gaussians by tail efficiency, rather than merely exposing 64 independent decisions.

Workload balancing does what it is designed to do—it lowers region-workload CV from 0.684 to 0.047—but does not add latency gain at matched quality: count-balanced risk strata reach $1.1034\times$ versus the full method's $1.1032\times$; their direct ratio is $0.9998\times$ with a confidence interval that crosses one. We therefore treat workload balancing as a conditioning and region-resolution regularizer, not as an independent source of speedup.

Table 7: **Region-count ablation on 2DGS, all thirteen scenes.** All variants use the same frozen 30k-iteration 2DGS checkpoints at native resolution, the same 16 calibration and 32 disjoint selection views, the 11-level ladder, $S_{\text{eff}} = \min(8, K_{\text{reg}})$ risk strata, and 48 Lagrangian penalties with the same proposal-generation procedure. Selection requires minimum-view teacher PSNR ≥ 48 dB and worst-view p99 ≤ 0.012; these are common constraints, not exact matching of attained distortion. The baseline is the 2DGS path with the exact $1/255$ opacity bound, so the speedups exclude the prior 3σ-to-exact-bound transition. Quality deltas and pair ratio R are relative to this baseline on the full test views. Each measurement batch has five arms with cached frozen activations and fifteen balanced same-process timing cycles. The added low-K_{reg} and high-K_{reg} batches each include the same frozen $K_{\text{reg}} = 1$ and 64 controls and the exact-bound baseline; each ratio uses its own batch's timing. Speedups are scene geometric means; quality deltas, R, and compilation times are arithmetic scene means. Brackets are 95% scene-bootstrap intervals. Compilation time excludes checkpoint loading.

K_{reg}	ΔPSNR	ΔSSIM	ΔLPIPS↓	R	vs. exact bound	vs. $K_{\text{reg}} = 1$ [95% CI]	compile (s)
1	-0.0049	-1.21×10^{-4}	-7.07×10^{-5}	0.9161	1.0838×	1.0000×	11.7
2	-0.0056	-1.34×10^{-4}	-8.94×10^{-5}	0.9027	1.0974×	1.0144× [1.0079, 1.0218]	18.0
4	-0.0068	-1.61×10^{-4}	-1.00×10^{-4}	0.8882	1.1129×	1.0287× [1.0169, 1.0432]	24.8
8	-0.0075	-1.67×10^{-4}	-1.11×10^{-4}	0.8801	1.1200×	1.0334× [1.0197, 1.0485]	37.7
64	-0.0093	-1.83×10^{-4}	-1.06×10^{-4}	0.8787	1.1216×	1.0349× [1.0215, 1.0496]	53.5
256	-0.0098	-1.93×10^{-4}	-1.09×10^{-4}	0.8782	1.1228×	1.0360× [1.0214, 1.0520]	57.9
512	-0.0092	-1.70×10^{-4}	-1.11×10^{-4}	0.8797	1.1186×	1.0355× [1.0232, 1.0504]	60.5
1024	-0.0104	-1.90×10^{-4}	-1.06×10^{-4}	0.8786	1.1188×	1.0357× [1.0220, 1.0516]	62.6

Table 8: **Region-construction ablation on all thirteen Speedy-Splat scenes.** Every row has $K_{\text{reg}} = 64$ decisions and uses the same probes, ladder, and search. Each validation-selected Pareto curve is interpolated in teacher MSE to 55 dB; only its two bracketing policies are timed in a common balanced process. Random preserves only work balancing; Uniform grid preserves only spatial locality; No-risk preserves Morton locality and work balancing; No-workload retains risk strata and Morton order but splits by Gaussian count. Work CV is the coefficient of variation of baseline pair workload across regions. Full/row above one means Full is faster; brackets are 95% scene-bootstrap intervals.

Grouping	Risk	Spatial	Work-bal.	Work CV	R	vs. AccuTile	Full/row [95% CI] (wins)
Random	no	no	yes	0.005	0.8832	1.0649×	1.0360× [1.0285, 1.0458] (13/13)
Uniform grid	no	yes	no	3.652	0.8823	1.0658×	1.0351× [1.0293, 1.0408] (13/13)
No risk strata	no	yes	yes	0.005	0.8480	1.0820×	1.0196× [1.0151, 1.0243] (13/13)
No workload balance	yes	yes	no	0.684	0.8087	1.1034×	0.9998× [0.9985, 1.0011] (6/13)
Full	yes	yes	yes	0.047	0.8076	1.1032×	–

D.5 WHY NOT SIMPLY RAISE THE CONSTANT?

The most direct objection to this paper is that our gain merely reflects $\tau_0 = 1/255$ being too conservative, and that setting it to $2/255$ or $4/255$ would capture it. We answer that first, because it determines what the rest of the section is worth.

Table 9 applies each fixed multiple of τ_0 to all thirteen scenes. Read by the mean, the objection looks correct: at $4\tau_0$ the average costs only -0.085 dB for $1.111\times$. Read by the worst scene it does not: the same $4\tau_0$ costs -0.426 dB on *bonsai*. And *bonsai* is the least favourable scene at *every* multiple, so a single constant is not free to sit at its own average — it must be clamped to whichever scene happens to be hardest.

Making that clamp explicit answers the objection quantitatively. A fixed constant matched to our worst-scene quality (-0.022 dB) can only reach $1.62\tau_0$ and $1.036\times$, against our $1.1009\times$ in this dedicated sweep. A fixed constant matched to that *speed* must run at $3.6\tau_0$, where the worst scene loses 0.334 dB—$15\times$ our worst-case error. Note also that our own worst scene is *garden*, not *bonsai*: the compiled policy specifically relieves the scene that constrains the global rule.

There is a second, quieter point. Choosing $2\tau_0$ or $4\tau_0$ safely presupposes knowing which scenes tolerate it, and the per-scene optimum found by our selector ranges over 1.5–$3.0\tau_0$. Recovering

Table 9: "Why not simply raise the constant?" Each row applies one fixed multiple of τ_0 to all thirteen scenes in one dedicated cutoff sweep. The mean is reassuring and the worst scene is not: the same scene (*bonsai*) is the least favourable at every multiple, so a single constant must be clamped to it. Matching OURS-REGIONAL's worst-scene quality caps a fixed constant at $1.62\tau_0$ and $1.036\times$; matching its $1.1009\times$ speed in this sweep forces the constant to $3.6\tau_0$, where the worst scene loses -0.334 dB against -0.022 dB. The independent all-view twelve-cycle main retiming is $1.096\times$. Choosing safely requires measuring the scene—the compilation step already present in OURS-GLOBAL.

Threshold	mean ΔPSNR	worst ΔPSNR	worst scene	vs. Official-Fixed
$1.25\tau_0$	-0.0011	-0.0030	*bonsai*	$1.0194\times$
$1.5\tau_0$	-0.0038	-0.0126	*bonsai*	$1.0305\times$
$2\tau_0$	-0.0126	-0.0493	*bonsai*	$1.0530\times$
$3\tau_0$	-0.0425	-0.2005	*bonsai*	$1.0855\times$
$4\tau_0$	-0.0853	-0.4255	*bonsai*	$1.1113\times$
$6\tau_0$	-0.1900	-0.9404	*bonsai*	$1.1497\times$
$8\tau_0$	-0.2889	-1.3708	*bonsai*	$1.1767\times$
Ours-Regional	$\mathbf{-0.0074}$	$\mathbf{-0.0218}$	*garden*	$\mathbf{1.1009\times}$

Table 10: **Calibration-view sensitivity on all thirteen Speedy-Splat scenes.** Each family uses $K_{\text{reg}} = 64$ and 32 disjoint selection views; only the number of calibration views changes. Curves are interpolated to 55 dB teacher PSNR and their bracketing policies are timed together. Brackets bootstrap scenes and parentheses count wins.

Calibration views	ΔPSNR	R	vs. AccuTile [95% CI] (wins)	16/row [95% CI] (wins)	Compile (s)
4	-0.0095	0.8661	$1.0827\times$ [1.0579, 1.1079] (13/13)	$1.0036\times$ [0.9781, 1.0257] (9/13)	12.0
8	-0.0109	0.8342	$1.0774\times$ [1.0260, 1.1197] (11/13)	$1.0085\times$ [0.9831, 1.0358] (7/13)	15.4
16	-0.0099	0.8076	$1.0866\times$ [1.0542, 1.1160] (11/13)	–	9.0
32	-0.0074	0.8000	$1.0889\times$ [1.0523, 1.1238] (11/13)	$0.9979\times$ [0.9792, 1.0229] (3/13)	17.2

even that scalar requires scene measurements and rendered acceptance tests. We call the process compilation because it automatically emits a frozen deployment asset; it replaces manual threshold choice rather than claiming that no offline selection occurs.

Calibration-view sensitivity. Region count is swept on all thirteen scenes in Table 6; Table 10 likewise varies calibration views over $4/8/16/32$ while matching every curve at 55 dB. All four settings have aggregate intervals above one versus AccuTile, while every direct interval against the 16-view default crosses one. Increasing from 16 to 32 views lowers R from 0.808 to 0.800 but yields statistically indistinguishable latency, so we retain 16 rather than tune this value per scene. Across three independent random 16/32 calibration/selection splits, speedup is $1.0945 \times \pm 0.0010\times$, pair ratio is 0.8299 ± 0.0023, and mean PSNR change is -0.0085 ± 0.0012 dB (Table 16).

D.6 DENSE, FLOW-ALIGNED TEMPORAL EVALUATION

We construct a continuous path by inserting three SE(3)-interpolated cameras between every adjacent pair of official test poses. This yields 359 frames over *bicycle*, *bonsai*, and *counter*. For each method, let the pruning residual be its rendered image minus the unmodified AccuTile image. We estimate bidirectional RAFT-Small flow (Teed and Deng, 2020) between consecutive AccuTile frames, backward-warp the preceding residual, reject occlusions by forward–backward consistency, and measure the remaining residual change. The OURS-GLOBAL and OURS-REGIONAL policies are chosen closest to 55 dB using held-out training views; neither test images nor the interpolated path participate in selection.

Region adaptation lowers the geometric-mean aligned residual change to $0.789\times$ the OURS-GLOBAL value while also reducing the pair ratio from 0.873 to 0.775. Because lower per-frame error alone can lower this temporal quantity, we also normalize each policy's aligned change by its own frame RMSE. That ratio is $0.990\times$ overall: no aggregate temporal regression, but no claim of

Table 11: **Dense, flow-aligned temporal evaluation.** OURS-GLOBAL and OURS-REGIONAL are selected near 55 dB using held-out training views, before constructing the test-camera paths. We insert three SE(3)-interpolated views between adjacent test poses and use bidirectional RAFT-Small consistency to mask occlusions. E is flow-aligned residual-change RMSE ($\times 10^{-3}$); E_n divides by each policy's frame RMSE. Lower E and Regional/Global are better. The aggregate uses geometric means (valid flow is transition-weighted).

Scene	Frames	Valid flow	R Global	R Regional	E Global	E Regional	E_n Regional/Global
Bicycle	97	45.8%	0.922	0.815	2.700	1.798	0.926×
Bonsai	145	77.4%	0.898	0.772	2.081	1.373	0.998×
Counter	117	79.6%	0.803	0.738	1.386	1.549	1.050×
Aggregate	359	69.6%	0.873	0.775	1.982	1.564	0.990×

Table 12: OURS-GLOBAL and OURS-REGIONAL policies compiled on AccuTile and transferred unchanged into two official optimized rasterizers. FlashGS retains its precise intersection and scheduling optimizations. R and teacher PSNR are measured against each rasterizer's native path. Every row uses twelve balanced same-process timing cycles; no row recalibrates or fine-tunes either policy.

Rasterizer	Resolution	Scenes	R	Teacher PSNR	vs. Official-Fixed	vs. Ours-Global
SeeLe	Standard	13	0.8003	54.75	**1.0908**×	1.0437×
SeeLe	4K (3840 px)	13	0.7539	56.41	**1.2602**×	1.1233×
FlashGS	Standard	13	0.8205	52.52	**1.1214**×	1.0605×
FlashGS	4K (3840 px)	13	0.7723	50.38	**1.3166**×	1.1379×

an improvement either. The negative *counter* result ($1.050\times$ normalized) is retained in Table 11; the evidence supports aggregate stability on these paths, not a universal no-popping guarantee.

D.7 TRANSFER TO INDEPENDENTLY OPTIMIZED RASTERIZERS

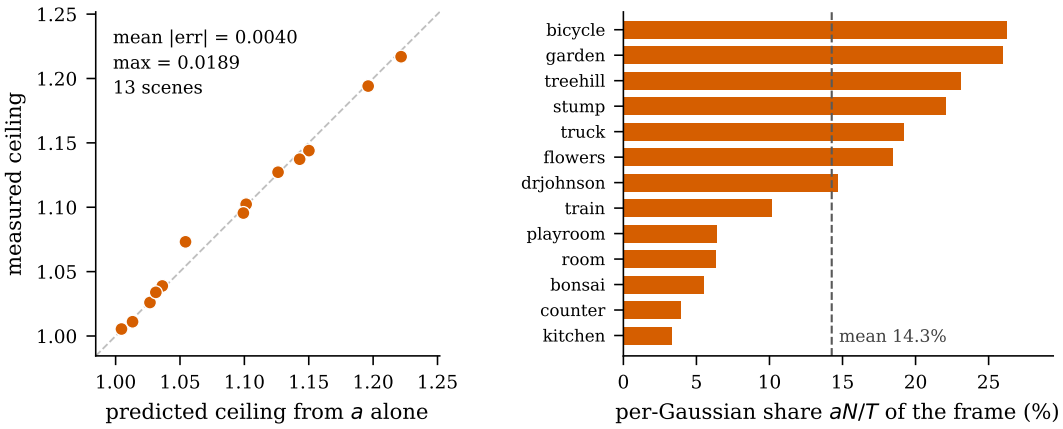
The strongest portability test is to run the Speedy-Splat policies underlying Table 1 inside software we did not write, without recalibration. We modify only the threshold lookup in the official SeeLe (Zhu et al., 2026) and FlashGS (Feng et al., 2025) rasterizers. SeeLe uses an axis-aligned footprint and a different blending path; FlashGS combines precise Gaussian intersection, adaptive scheduling, and pipelining.

Table 12 reports both transfers under the same twelve-cycle protocol as the main table. SeeLe becomes $1.091\times$ faster at standard dataset resolution and $1.260\times$ at 4K. FlashGS improves by $1.121\times$ and $1.317\times$, and the regional policy beats transferred OURS-GLOBAL by $1.061\times$ and $1.138\times$. The FlashGS result is especially useful because it shows the policy still removes work after a strong geometric tile-intersection optimization. These are transfer results, not claims against the full SeeLe camera-clustering pipeline or FlashGS's published end-to-end benchmark.

D.8 THE COST MODEL PREDICTS, AND ONE OF ITS PREDICTIONS IS NEGATIVE

Section B.1 fits an empirical affine model within controlled sweeps; here we test whether it is useful for dispatch. An oracle visible-subset comparison gives an estimated upper bound for the tested region pre-culler before implementing its runtime path. Across thirteen scenes, the estimate matches the realized ceiling to a mean absolute error of 0.004 (Figure 6a). The result is unfavourable: the per-Gaussian term is only 14.3% of the honest frame on average (3.3–26.3%), so the tested perfect, free oracle is capped near $1.17\times$ and the implemented culler— conservative predicate and planning included—returns $1.010\times$ overall. We report this as a negative result rather than omitting it, because it is the same model that produces the positive one.

The model also says *where* the negative result reverses. Dense stress-test checkpoints have a larger measured aN/T share, even though their pair count also changes. Table 13 and Figure 5b show that on the tested checkpoints above ∼2M Gaussians the two mechanisms compose nearly multiplicatively, reaching $1.235\times$ at standard dataset resolution, and the pre-culling arm is image-bit-exact in



(a) the model predicts the ceiling

(b) and says that ceiling is low

Figure 6: **A prediction made before the implementation, and then tested.** *(a)* Fitting only a — from a single oracle-subset comparison taking seconds — predicts the measured ceiling of region pre-culling to a mean absolute error of 0.0040 over thirteen scenes. *(b)* The prediction is unfavourable: the per-Gaussian term is 14.3% of the honest frame on average, so the tested perfect, free oracle is capped near $1.17\times$. The estimate is specific to this checkpoint family, resolution, GPU, and pre-culling design.

Table 13: Policy truncation primarily reduces pair-correlated work; image-bit-exact region pre-culling skips conservatively rejected Gaussian blocks before projection. Standard dataset resolution, planning cost included. This five-arm stress microbenchmark uses at most twelve evenly spaced test views per scene, so its policy column is not a replacement for the all-view twelve-cycle main-table timing. The pre-culling arm's maximum absolute pixel difference against the unmodified render is 0 in all 52 runs, so the quality cost of the combination is that of the policy alone. Legacy dense A/B are historical packaged checkpoints used only as density stress tests, not as evidence for a named model family. Below roughly 500K Gaussians the frustum test costs more than it saves and the dispatch rule of Section B.3 disables it.

Checkpoint	Mean Gaussians	Base (ms)	Policy	Pre-cull	**Both**	Gain from stacking
Legacy dense A	2,933,368	3.63	1.080×	1.056×	**1.142×**	1.058×
Legacy dense B	2,519,315	2.52	1.138×	1.080×	**1.235×**	1.085×
DashGaussian	2,167,600	2.47	1.144×	1.063×	**1.220×**	1.067×
Speedy-Splat	407,593	0.78	1.132×	0.988×	**1.115×**	0.984×

all 52 runs. Two rows use legacy packaged checkpoints and are deliberately not attributed to named model families. Below $\sim$500K Gaussians pre-culling costs more than it saves ($0.988\times$ on Speedy-Splat) and the dispatch rule of Section B.3 switches it off. At 4K the measured per-Gaussian share falls to 3–12% of the frame, so the planning-cost test selects the policy alone.

D.9 Cost

Compilation reads the checkpoint and calibration views, runs both probes, and solves selection in 9.0 s per scene on average for the default $K_{\text{reg}} = 64$ (at most 14.0 s, excluding checkpoint load). The sensitivity probe invokes backward-mode differentiation but updates no checkpoint parameter and needs no optimizer state. The exported policy is one `uint8` level per Gaussian plus a table of at most 256 thresholds: exactly N bytes for the indices and under 1 KB for the table. The compact Speedy-Splat checkpoints occupy 0.14–0.88 MB; dense ports scale linearly beyond that range. Inference reads the byte during tile enumeration, with no extra kernel or per-frame policy inference. The cost amortizes after roughly 16k rendered frames at 4K and 126k at standard dataset resolution.

Table 14: **Attribution behind the simplified backbone comparison in Table 1.** Exact-$1/255$ replaces a released 3σ tile box by the opacity-aware bound already established by prior work (Radl et al., 2024; Hanson et al., 2025a); we do not count that transition as a compiler contribution. $K_{\text{reg}} = 1$ and $K_{\text{reg}} = 64$ are our compiler with one and 64 decisions. Values are speedups over the official released path except for the final incremental column.

Backbone	Exact-$1/255$	Ours ($K_{\text{reg}} = 1$)	Ours ($K_{\text{reg}} = 64$)	K_{reg}=64/K_{reg}=1
Speedy-Splat	$1.000\times$	$1.048\times$	$1.095\times$	$1.047\times$
2DGS	$1.313\times$	$1.422\times$	$1.474\times$	$1.037\times$
Scaffold-GS	$1.252\times$	$1.301\times$	$1.320\times$	$1.014\times$

D.10 Limitations

Calibration needs the training views. Compilation measures pair savings and tail energy by rendering, so it needs the camera poses and images the checkpoint was fitted to. It is a per-scene calibration step, not a data-free transform of an arbitrary released `.ply`. It does not need the optimizer state or the training run, and it performs no parameter update.

Scope is threshold-based tile enumeration. The method assumes a rasterizer whose footprint is governed by a contribution threshold. It applies directly to AccuTile-style rasterizers, to SeeLe, and to 2DGS, Scaffold-GS, or vanilla 3DGS once their 3σ box is replaced by such a bound. Oriented-box, convex-hull and non-tiled rasterizers expose a different control surface and are out of scope.

Regions do not beat primitives. As Table 2 shows, K_{reg} region-shared thresholds tie per-Gaussian control on speed. They reduce the number of independent search decisions and may regularize selection, but our current packed format stores one byte per Gaussian for either policy. We make no compression or superiority claim for regional control.

Static policies and camera motion. The exported policy is fixed. Our dense flow-aligned test covers 359 frames on three scenes and has no aggregate regression, but *counter* regresses by 5.0% after normalization. Broader trajectories and a perceptual user study would be needed for a universal no-popping claim.

Held-out constraints are not test guarantees. Three random calibration/selection splits produce nearly identical aggregate speed and pair ratios, but the per-Gaussian multiplier vectors agree exactly only 49.5% of the time. On unseen test views, 18 of the 39 split–scene runs exceed the 0.012 selection cap, eight reach at least 0.015, and the maximum is 0.032. The cap controls held-out training views; it does not certify every unseen pose. Selection and test diagnostics both use floating-point residuals and the same $4\times$ spatial subsample, so the excesses are not a dtype mismatch.

Tile-aligned residual structure. Lossy support truncation removes a Gaussian from a whole tile at once. In a three-scene, 92-test-view diagnostic, the residual finite-difference at the true 16-pixel tile phase is the largest of all 16 phases in every view. Its mean amplitude is only 0.19–0.31 of one 8-bit code value, but its mean boundary/non-boundary ratio is 5.86–$6.18\times$ (Section I). This establishes structured error, not perceptual visibility; a controlled user study remains absent.

One GPU class. All timings are from RTX 5000 Ada GPUs; the four locally available devices are the same model. The cost model suggests the balance between the two mechanisms shifts on hardware with a different memory-to-compute ratio, but a second desktop or mobile hardware class remains unmeasured.

E Backbone-transition attribution

The main table intentionally presents the deployment-facing comparison "official model versus model + Ours." Table 14 provides the accounting needed to interpret that compact presentation. Speedy-Splat already exposes the exact $1/255$ footprint, so its full gain is due to our compiler. The public 2DGS and Scaffold-GS rasterizers start from 3σ; installing an exact opacity-aware bound contributes the first speedup stage and is prior work, while $K_{\text{reg}} = 1$ and $K_{\text{reg}} = 64$ are our compiler stages.

E.1 Threshold semantics and low-opacity Gaussians

The exported lookup table stores absolute thresholds $m_g\tau_0$. The default Speedy-Splat path uses these thresholds directly, without an opacity-relative cap; a threshold at or above Gaussian opacity yields no tile pairs. The 2DGS and Scaffold-GS threshold-bound ports likewise reject primitives whose opacity cannot exceed the selected threshold. This changes their participation in a render, not the stored checkpoint parameters. The FastGS port instead applies $\min(m_g\tau_0, 0.95\alpha_g)$ and subtracts 10^{-6} when the cap binds before constructing its backend-specific extent. Thus preserving nonempty support is not a method-wide invariant, and an all-one policy should not be assumed to reproduce every backend's native pair list merely from its multiplier values.

A synthetic check of the default Speedy-Splat path used five opacities spanning $0.5\tau_0$ to 0.5. Both scalar and packed all-one thresholds reproduced the baseline images bit-for-bit and preserved pair counts in these cases. For an opacity of $0.5\tau_0$, the baseline and both all-one paths emitted zero pairs; the capped FastGS scalar and packed all-one paths emitted four pairs versus zero for their native baseline, while pixel-level alpha rejection kept all three images identical. This distinguishes image equivalence from work equivalence; it is a bounded regression check, not a universal floating-point proof.

F The checkpoint families used by Fragment Pruning

Fragment Pruning (Ye et al., 2024) reports its learned per-Gaussian fragment cutoff on vanilla 3DGS and Mini-Splatting checkpoints. We therefore apply TileSkipper independently to clean official-style 30k 3DGS checkpoints and official Mini-Splatting outputs over the same thirteen scenes. This is a checkpoint-composition test, not an implementation of Fragment Pruning and not a cross-hardware speed contest: their published FPS uses a Jetson Orin NX and an OpenGL renderer, whereas Table 15 uses our CUDA rasterizer on RTX 5000 Ada.

At standard dataset resolution, the deployment-facing totals are $1.493\times$ for 3DGS and $1.346\times$ for Mini-Splatting; at fixed 1080p they rise to $1.841\times$ and $1.599\times$, with all thirteen scenes faster in both settings. Those totals must not be attributed entirely to our compiler: at standard resolution, installing the prior exact $1/255$ bound already gives $1.454\times$ and $1.266\times$, respectively. The regional policy contributes a further $1.027\times$ and $1.063\times$ ($1.038\times$ and $1.089\times$ at 1080p). Thus the experiment supports compatibility with dense and aggressively pruned checkpoints, but it does not establish superiority to Fragment Pruning's learned cutoff. Their reported dataset-level ratios are in a similar numerical range, but the hardware, renderer, and 60-FPS cap make direct ranking invalid.

Every selected policy passes the 48-dB and 0.012 p99 constraints on the disjoint held-out training views used by the compiler. On unseen test views, mean ground-truth PSNR changes remain only $-0.010/-0.014$ dB for 3DGS/Mini-Splatting, but at fixed 1080p Mini-Splatting's worst test view reaches 45.76 dB teacher PSNR and 0.0259 p99. This is a concrete instance of the held-out-versus-test limitation in Section D.10; we report it rather than treating the selection constraint as a certificate.

G Calibration-view split stability

To test whether the compiler depends on one favorable view partition, we repeat the full $K_{\mathrm{reg}} = 64$ Speedy-Splat experiment with random seeds 17, 43, and 89. Each seed independently samples 16 calibration and 32 disjoint selection views in every scene; the ladder, probes, constraints, and search are unchanged. The three policies are then evaluated and timed with AccuTile in the same four-arm process. Table 16 shows only $0.0010\times$ standard deviation in the thirteen-scene speedup, and all three splits improve all thirteen scenes. The exact per-Gaussian multiplier agreement across split pairs is 49.5%: different view samples find different near-equivalent policies, while their aggregate distortion, work, and latency remain stable. Mean compilation time is 9.1 seconds per scene. This aggregate stability does not extend to the test-view tail constraint: 18 of 39 split–scene runs exceed p99 0.012, eight reach at least 0.015, and the maximum is 0.032.

A scalar safety margin helps but does not solve view transfer. We reselect from each split's already-rendered candidate table under validation p99 caps of 0.010 and 0.008, without consulting

Table 15: **TileSkipper on the two checkpoint families used by Fragment Pruning.** Standard-resolution and fixed 1920×1080 rendering on RTX 5000 Ada; each row uses the same patched vanilla-3DGS CUDA rasterizer and twelve balanced timing cycles. "Base → +Ours" is the deployment-facing total; "AccuTile → +Ours" isolates our compiled policy after the prior $3\sigma \rightarrow 1/255$ exact-bound transition. Intervals bootstrap scenes and parentheses count wins. These are not head-to-head Jetson/OpenGL measurements against Fragment Pruning.

Checkpoint	Resolution	Dataset	Base → +Ours [95% CI] (wins)	AccuTile → +Ours	3σ →AccuTile	ΔPSNR
3DGS	standard 1080p	All 13	$\mathbf{1.493}\times$ [1.350, 1.656] (13/13)	$\mathbf{1.027}\times$ [1.014, 1.042] (13/13)	$1.454\times$	−0.003
		Mip-NeRF 360	$1.729\times$ [1.532, 1.959] (9/9)	$1.042\times$ [1.023, 1.064] (9/9)	$1.659\times$	−0.014
		Tanks&Temples	$2.246\times$ [2.013, 2.506] (2/2)	$1.027\times$ [1.020, 1.035] (2/2)	$2.186\times$	−0.009
		Deep Blending	$2.003\times$ [1.947, 2.060] (2/2)	$1.028\times$ [1.021, 1.036] (2/2)	$1.948\times$	+0.003
		All 13	$\mathbf{1.841}\times$ [1.657, 2.035] (13/13)	$\mathbf{1.038}\times$ [1.024, 1.054] (13/13)	$1.774\times$	−0.010
Mini-Splatting	standard 1080p	All 13	$\mathbf{1.346}\times$ [1.260, 1.451] (13/13)	$\mathbf{1.063}\times$ [1.046, 1.081] (13/13)	$1.266\times$	−0.008
		Mip-NeRF 360	$1.593\times$ [1.477, 1.733] (9/9)	$1.090\times$ [1.069, 1.115] (9/9)	$1.462\times$	−0.017
		Tanks&Temples	$1.770\times$ [1.748, 1.792] (2/2)	$1.081\times$ [1.071, 1.091] (2/2)	$1.637\times$	−0.013
		Deep Blending	$1.473\times$ [1.410, 1.538] (2/2)	$1.095\times$ [1.077, 1.114] (2/2)	$1.345\times$	−0.004
		All 13	$\mathbf{1.599}\times$ [1.505, 1.703] (13/13)	$\mathbf{1.089}\times$ [1.074, 1.107] (13/13)	$1.468\times$	−0.014

Table 16: **Calibration/selection split stability on all thirteen Speedy-Splat scenes.** Each row independently samples 16 calibration and 32 disjoint selection views; $K_{\text{reg}} = 64$ and all compiler settings are fixed. Speedup intervals bootstrap scenes.

Split seed	ΔPSNR	R	vs. AccuTile	Wins	Compile (s)
17	−0.0097	0.8325	$1.0934\times$ [1.0709, 1.1176]	13/13	9.1
43	−0.0073	0.8283	$1.0947\times$ [1.0752, 1.1160]	13/13	9.1
89	−0.0084	0.8288	$1.0953\times$ [1.0762, 1.1178]	13/13	9.1

test views. As Table 17 shows, the tighter caps reduce violations from 18 to 5 and 3 of 39 runs, respectively, while increasing the retained-pair ratio from 0.830 to 0.877. All three failures at 0.008 are `kitchen`; its worst p99 remains 0.021–0.030 across the three splits. Thus a global margin trades work for robustness but does not address a view-coverage failure.

H Spatial heterogeneity of safe global thresholds

Figure 7 provides the pixel-level evidence behind the Introduction's claim that a single global threshold is forced into a compromise: it is too loose on easy regions and too tight on hard ones. We partition a representative *playroom* view into a coarse spatial grid; for each region we sweep a global threshold $\tau \in \{1, 2, 4, 8, 16\}/255$ and report the largest $\tau^\star$ whose local PSNR (computed over the region's pixels only) stays within 0.03 dB of the $1/255$ baseline. Each region is colored by its $\tau^\star$. The grid contains the full $16\times$ spread from $1/255$ to $16/255$ on a *single view*. The four callouts are selected at distinct values: region 1 is a low-detail example at $16/255$, region 2 a material patch at $8/255$, region 3 a fine-detail example at $4/255$, and region 4 an edge or boundary at $2/255$. Cells at $1/255$ appear in the map but are not among these four callouts. This diagnostic establishes local heterogeneity; it does not by itself measure how much of that heterogeneity a particular compiled partition captures. Indeed, *playroom* is one of two scenes whose selected Regional policy collapses to a single level; pixel-local heterogeneity is therefore not sufficient evidence of deployable regional gain.

I Tile-boundary residual structure

Tile support is a discrete action: when a tail no longer intersects a tile, all of its fragments in that tile disappear together. We therefore test for the specific grid artifact that aggregate PSNR, SSIM, and LPIPS may hide. On `bicycle`, `bonsai`, and `counter`, we choose real Global, Regional, and per-Gaussian policies nearest 55 dB using validation metadata only, then render all 92 official test views. For each candidate-minus-backbone residual, we measure the RGB finite-difference magnitude at every pixel phase modulo 16.

Table 17: **Validation-cap guard-band diagnostic over the same 39 randomized split–scene candidate tables.** Policies are reselected using only their held-out training-view measurements and then evaluated on official test views. The original 0.012 row reproduces the reported policies. A tighter validation cap reduces unseen-view violations but also retains more pairs; it does not remove the persistent `kitchen` failures. This post-hoc diagnostic is not used to select the paper's primary operating point.

Validation p99 cap	Test p99 > 0.012	Test p99 ≥ 0.015	Worst test p99	Mean R	Mean worst-view teacher PSNR
0.012	$18/39$	$8/39$	0.0320	0.830	52.06 dB
0.010	$5/39$	$5/39$	0.0305	0.851	54.08 dB
0.008	$3/39$	$3/39$	0.0301	0.877	56.43 dB

(a) Largest safe global threshold by image region

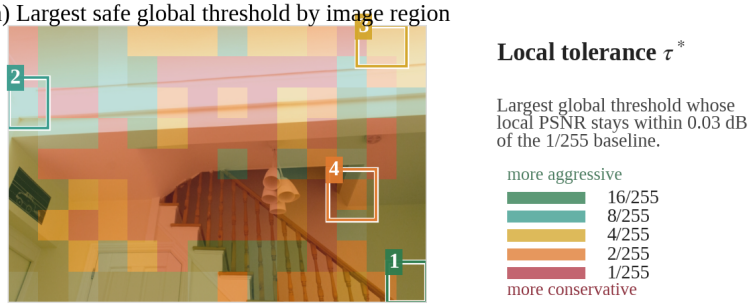


Figure 7: **Largest safe global tile threshold varies by $16\times$ across regions of a single view.** Each grid cell of a representative *playroom* render is colored by the largest $\tau^\star \in \{1, 2, 4, 8, 16\}/255$ whose local PSNR stays within 0.03 dB of the $1/255$ baseline. The four highlighted examples are at **(1)** $16/255$ (low detail), **(2)** $8/255$ (material patch), **(3)** $4/255$ (fine detail), and **(4)** $2/255$ (edge or boundary). Other grid cells include the conservative $1/255$ value, giving the reported full spread. This is a local tolerance diagnostic rather than an evaluation of the compiled partition.

As shown in Figure 8, the true tile boundary (phase zero) is the largest of the 16 phases in all 92 views for every policy. The pooled mean boundary/non-boundary ratios are $5.86\times$, $6.18\times$, and $5.93\times$ for Global, Regional, and per-Gaussian control, respectively. Their absolute mean boundary jumps are 0.192, 0.287, and 0.312 of one 8-bit code value. Thus lossy tile-support truncation does create grid-aligned residual structure, but this diagnostic alone does not establish that it is visible at normal magnification. The similar result for all three controls also shows that the effect is tied to the tile-level action rather than regional grouping specifically.

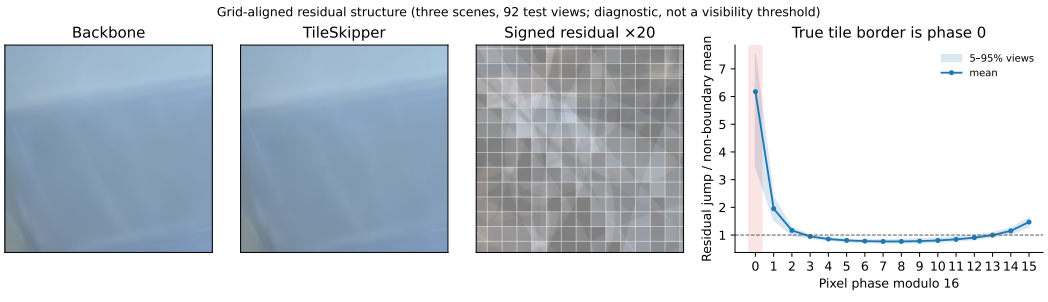


Figure 8: **Grid-aligned residual diagnostic.** The crop is selected by maximum boundary-weighted residual energy and the signed residual is amplified $20\times$; white lines show the true tile grid. The phase plot pools the Regional policy over three scenes and 92 test views. The highlighted phase zero is the actual tile border. The plot diagnoses residual structure and is not a perceptual visibility threshold.

J QUALITATIVE COMPARISON

Figure 9 shows the backbone and the compiled policy on four held-out `kitchen` test views. For each view we crop the 160×160 window where the two renders differ *most*, so the figure shows the hardest region rather than a representative one, and amplify the difference by $20\times$ to make it visible at all. Per-view PSNR against ground truth changes by $+0.00$ to -0.02 dB and the largest single-pixel difference is 0.074 on a $[0, 1]$ scale—roughly 19 quantization steps of an 8-bit channel, concentrated on the woven placemat and the specular edge of the window frame. Their concentration in high-frequency content shows where even a conservative allocation remains vulnerable; it should not be read as evidence that the compiler intentionally assigns a looser cutoff to those regions.

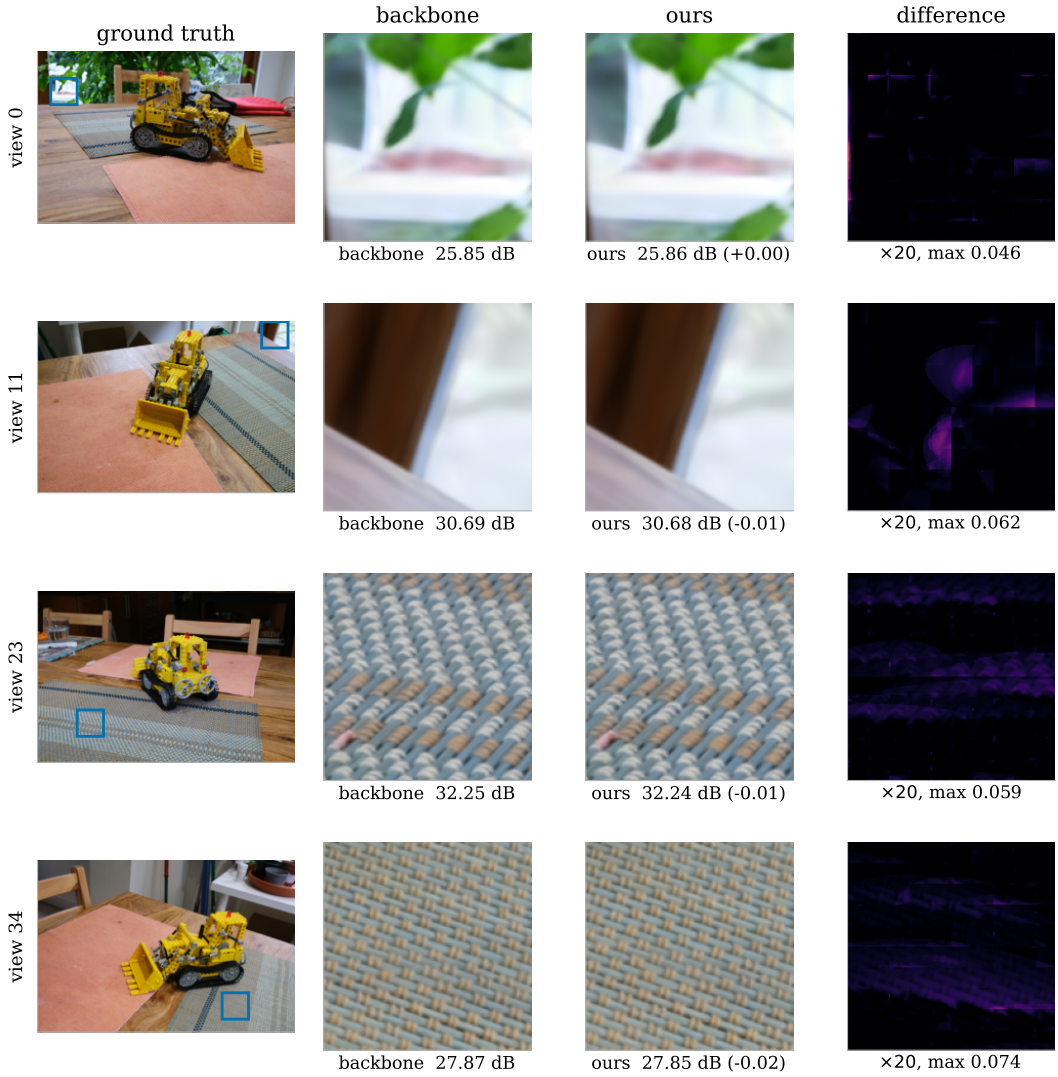


Figure 9: **Backbone versus the compiled policy on held-out `kitchen` views.** Crops are placed automatically at the window of largest disagreement, not chosen by hand; the difference column is amplified $20\times$. PSNR is against ground truth, so it is comparable to Table 1.