PRUNE BY PROXY: A UNIFIED FRAMEWORK FOR 3D GAUSSIAN SPLATTING PRUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-hoc pruning makes 3D Gaussian Splatting (3DGS) compact by removing primitives ranked by an importance score. Existing scores rank Gaussians by their attributes, rendering contributions, or parameter sensitivity, yet what pruning actually trades off is the photometric cost of deleting a Gaussian. Evaluating this deletion cost by leave-one-out rendering requires one re-rendering per Gaussian, and no common target or protocol exists to tell which of the tractable scores actually estimate it. We present *Prune by Proxy*, a framework that fixes the deletion cost as the common target, specifies every score by its scoring information, scored attributes, and aggregation rule, and thereby transfers the zero-cost proxies of NAS-Bench-Suite-Zero from network pruning and architecture search to 3DGS. Within this framework we show that a multiplicative gate on activated opacity is the coordinate of deletion: alpha blending is affine in a single gate, so the deletion-induced image change equals minus the gate derivative, and the gap between a first-order estimate and the exact deletion cost lies entirely in the photometric loss, where it has a closed form. The resulting Corrected L_1 and L_2 scores equal the exact single-Gaussian deletion cost in the ideal blending model and cost one forward and one backward pass per view. At 90% pruning on Mip-NeRF 360, Tanks & Temples, and Deep Blending, Corrected L_1 ranks first in eight of nine dataset–metric comparisons and exceeds PUP on all thirteen scenes by 0.58 dB on average at one-third of its scoring time. The closed-form correction alone adds up to 1.6 dB over its first-order control, and seven transferred proxies also surpass all published 3DGS scores, with deletion alignment explaining which proxies transfer.

1 INTRODUCTION

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) represents scenes with millions of explicit primitives, which makes storage and memory costly. Post-hoc pruning removes Gaussians from a trained scene ranked by an importance score, optionally followed by refinement. By the information they use, existing scores fall into three categories: attribute-based methods derive importance from Gaussian properties or local attribute context (Kerbl et al., 2023; Lee et al., 2024; Yang et al., 2026); contribution-based methods accumulate rendering statistics such as visibility, transmittance, or blending weights across views (Fan et al., 2024; Girish et al., 2024; Fang & Wang, 2024; Niemeyer et al., 2025); and sensitivity-based methods measure the response of the reconstruction objective to perturbations of Gaussian parameters (Hanson et al., 2025). Each captures a different notion of importance.

What pruning actually changes, however, is the reconstruction error: the deletion cost of a Gaussian is the error increase caused by removing it. Direct evaluation requires a leave-one-out re-rendering per primitive, which is infeasible for millions of Gaussians, so practical scores can only act as proxies of it. Yet no existing score is formulated against this target, and without a common target the scores cannot be systematically interpreted, compared, or redesigned as deletion-cost estimators.

We propose *Prune by Proxy*, a framework that fixes the single-Gaussian deletion cost as the common target and makes every score explicit through three choices: the scoring information, the scored attributes, and the aggregation rule. It transfers zero-cost proxies from network pruning and architecture search to 3DGS by substituting the ranked unit, the query, and the forward model with the

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.

Gaussian, the camera, and alpha blending, so that transferred proxies and published Gaussian scores become instances of the same construction.

The framework also shows how to make a derivative-based score exact. Following the multiplicative gates of SNIP (Lee et al., 2019), we attach a scalar gate to activated opacity: setting it to zero removes the Gaussian, and the rendered image is affine in it. The gap between the gate derivative and the finite deletion cost therefore lies entirely in the photometric loss, where it has a closed form. Adding this remainder yields loss-corrected scores that equal the exact deletion cost in the ideal blending model, at the price of one forward and one backward pass per view.

Our contributions are fourfold:

- A unified framework for post-hoc Gaussian pruning that uses the single-Gaussian deletion cost as a common target and makes score construction explicit through scoring information, scored attributes, and the aggregation rule;
- A transfer of zero-cost proxies to 3DGS pruning: thirteen proxies from NAS-Bench-Suite-Zero and ZiCo are rebuilt as per-Gaussian scores within the framework; seven of them surpass all published 3DGS scores, and their deletion alignment explains which proxies transfer;
- A theoretical analysis that identifies the opacity gate as a deletion-aligned coordinate and establishes the relation between derivative-based estimates and finite deletion costs under alpha-blending assumptions;
- A family of loss-corrected deletion scores that augment signed first-order estimates with the closed-form photometric remainder, yielding scores aligned with the deletion cost.

Under a shared evaluation protocol on Mip-NeRF 360, Tanks & Temples, and Deep Blending at 90% pruning, Corrected L_1 ranks first in eight of nine dataset–metric comparisons and exceeds PUP on all thirteen scenes by 0.58 dB on average at one-third of its scoring time.

2 Related Work

Gaussian pruning and compression. Post-hoc pruning ranks the Gaussians of a trained scene, removes the least important ones, and optionally refines the rest (Fan et al., 2024; Yang et al., 2026; Hanson et al., 2025; Fang & Wang, 2024; Niemeyer et al., 2025). Complementary compression methods quantize and entropy-code Gaussian attributes, often after removing redundant primitives (Niedermayr et al., 2024; Chen et al., 2024; Girish et al., 2024), or reorganize the unstructured Gaussian set into compact structures such as anchors (Lu et al., 2024); recent systems combine pruning, alternative primitives, compact representations, and rendering-aware optimization (Wang et al., 2026; Liu et al., 2026). Such combined gains obscure the contribution of the selection criterion itself; we therefore compare scores within one fixed post-hoc pruning pipeline.

Gaussian importance scores. Existing scores fall into three groups by the information they read. Attribute-based scores use intrinsic properties or local attribute context, from opacity heuristics to neighborhood-aware predictors (Kerbl et al., 2023; Lee et al., 2024; Yang et al., 2026); contribution-based scores accumulate view-dependent rendering statistics such as visibility, transmittance, or blending weights (Fan et al., 2024; Girish et al., 2024; Fang & Wang, 2024; Niemeyer et al., 2025); and sensitivity-based scores measure the response of the reconstruction objective to parameter perturbations, as in the curvature-based criterion of PUP (Hanson et al., 2025). These criteria differ in construction, yet each implicitly estimates how removing a Gaussian affects the reconstruction. None of them is explicitly defined with respect to this quantity, leaving no common reference for interpretation and systematic comparison. We make this target explicit as the single-Gaussian deletion cost, defined as the change in reconstruction loss caused by removing one Gaussian, and formulate these scores as proxies of it.

Zero-cost proxies. Zero-cost proxies predict the trained performance of a network, or the importance of its weights, from parameters, gradients, activations, or input responses at negligible cost (Abdelfattah et al., 2021; Krishnakumar et al., 2022). Transferring them to Gaussian pruning changes both the ranked object and the target: a proxy must estimate the error increase caused by deleting a primitive rather than the accuracy of an architecture. We make this transfer explicit by

Table 1: Slot substitution. The scoring schema is unchanged; only the fill changes from the weight space to a Gaussian scene.

Slot	Network schema	3DGS fill
Ranked unit	weight / channel / cell	3D Gaussian $g_i \in \mathcal{G}$
Query	input $\mathbf{x}$	camera ϕ_v, optionally with I_v
Forward model	layer products	alpha blending along a ray

fixing the deletion cost as the common target and specifying each score by its scoring information, scored attributes, and aggregation rule. This yields a unified view of published Gaussian scores and transferred proxies, and measuring them against deletion exposes the gap that first-order derivative scores ignore between an infinitesimal perturbation and actual deletion, which our loss-corrected scores close.

3 A Unified Proxy Formulation for Gaussian Pruning

Existing Gaussian pruning scores lack a common target. This section fixes one and uses it to bring zero-cost proxies from network pruning and architecture search into 3DGS. Section 3.1 defines the target, the single-Gaussian deletion cost, whose exact evaluation requires one leave-one-out rendering per Gaussian and is therefore approximated by tractable proxies. Section 3.2 transfers the zero-cost proxy setting to a trained Gaussian scene by substituting the ranked unit, the query, and the forward model. Section 3.3 specifies every construction, transferred or published, by three choices: the scoring information, the scored attributes, and the aggregation rule. Transferred proxies and published 3DGS scores thereby become directly comparable estimators of the same quantity (Table 2).

3.1 Reference deletion objective

Let $\mathcal{G} = \{g_i\}_{i=1}^N$ be a trained Gaussian scene with $g_i = (\boldsymbol{\mu}_i, \mathbf{s}_i, \mathbf{r}_i, \mathbf{c}_i, \alpha_i)$: position, activated scale, rotation, color, and activated opacity. For a binary mask $\mathbf{m} \in \{0,1\}^N$, $R_v(\mathbf{m})$ denotes the image rendered from camera ϕ_v by the retained Gaussians $\{g_i : m_i = 1\}$; $\mathbf{m} = \mathbf{1}$ is the full scene, and $\mathbf{1} - \mathbf{e}_i$, with $\mathbf{e}_i$ the ith unit vector, deletes Gaussian i. For a scalar objective J of the rendered images, the single-Gaussian deletion cost is

$$\Delta J_i = J(\mathbf{1} - \mathbf{e}_i) - J(\mathbf{1}). \tag{1}$$

A positive ΔJ_i means that deleting Gaussian i increases the objective; an ideal score ranks Gaussians by ΔJ_i and removes those whose deletion increases the objective least.

Reference objective. We instantiate J with the L_1 term of the 3DGS training loss on the V training views, measured before refinement:

$$\mathcal{E}(\mathbf{m}) = \sum_{v=1}^{V} \ell_{1,v}(\mathbf{m}), \qquad \ell_{1,v}(\mathbf{m}) = \big\| R_v(\mathbf{m}) - I_v \big\|_1, \tag{2}$$

where I_v is the reference image of view v and the norm sums over pixels and color channels. The reference deletion cost $\Delta\mathcal{E}_i$, i.e., equation 1 with $J = \mathcal{E}$, is the common target that every score in this paper approximates, whether or not it reads I_v. Joint deletion, the actual pruning operation, extends this single-Gaussian target (Appendix B).

3.2 Transfer setting

Zero-cost proxies score network units or whole architectures from a few forward and backward passes, without training: network-pruning criteria rank weights or channels, and NAS proxies rank candidate architectures. Transferring them to a trained Gaussian scene keeps **the scoring schema unchanged; its slots are filled with 3DGS objects** (Table 1).

A photometric score additionally reads the reference image I_v; a camera-perturbation score compares responses from nearby poses; and direct-attribute scores such as L2-GS need neither a camera

nor rendering. Alpha blending couples a primitive's color contribution to its occlusion of later contributions, which Section 4 exploits. All constructions then share the same selection rule: extract per-Gaussian responses, aggregate them into a score, and retain the K Gaussians with the highest scores. Fixing the slots still leaves three choices free: two refine what is read from the filled slots (the information read at the query, and the attributes scored on the ranked unit), and one fixes the aggregation step of the selection rule. Section 3.3 specifies these choices, and Table 2 records them for every method.

3.3 Proxy construction

The slot substitution of Table 1 fixes the types of the objects every construction operates on; the three choices below specify what remains free.

Scoring information. This choice fixes what information the score reads from each view. We distinguish photometric information requiring reference images from render-only information available without them. For scalar-objective methods, J_v specifies the differentiated quantity: the L_1 error provides a per-view term of $\mathcal{E}$, and the rendered RGB sum defines an alternative objective; other methods differentiate individual image outputs or camera-induced image changes.

Scored attributes. This choice fixes which attributes of each Gaussian the score is computed from. Derivative-based scores differentiate either the stored spatial parameters $(\boldsymbol{\mu}_i, \boldsymbol{\rho}_i)$, with $\boldsymbol{\rho}_i = \log \mathbf{s}_i$, or a multiplicative gate attached to activated opacity; Section 4.1 establishes the relation of these two coordinates to deletion. Direct-statistic scores instead read descriptors such as opacity or footprint (Table 2).

Aggregation. This choice fixes how the extracted responses are pooled into a score. Sums of absolute values or squares, gradient statistics, and Gram scalarizations can produce different scores from the same information and scored attributes (Tables 2 and 4).

Adapting a proxy. Each adapted proxy keeps the response of its source criterion and changes only what the response is computed on. Parameter-level criteria (SNIP, Fisher, GradNorm, GraSP, SynFlow, and Plain) replace a network weight by a Gaussian attribute or its opacity gate and the training loss by J_v, and sum the per-Gaussian response over views. Architecture-level proxies score a network from statistics across an input batch; with the training views as the batch, they become per-Gaussian statistics: cross-view variation of normalized gradients (Jacov), the ratio of gradient mean to deviation across views (ZiCo), binary activity codes across views (NASWOT), and responses to camera rather than input perturbations (Zen and EPE). Cost and norm baselines map to rasterization usage (flops) and opacity magnitude (l2-norm). Table 2 records the resulting constructions, and Appendix E gives their exact definitions.

Table 2 covers the zero-cost proxies of NAS-Bench-Suite-Zero (Krishnakumar et al., 2022), ZiCo (Li et al., 2023), published 3DGS scores, and the two corrected scores of Section 4.3; Appendix E.1 lists the original NAS proxy categories.

4 Opacity-Gate Deletion Analysis and Loss-Corrected Scoring

Section 3 casts every pruning score as a proxy of the deletion cost. This section shows that in 3DGS the deletion cost itself can be obtained at the cost of a gradient. Three facts combine. First, a multiplicative gate on activated opacity expresses deletion, and alpha blending is affine in a single gate, so the deletion-induced image change equals minus the gate derivative (Section 4.1). Second, the renderer therefore contributes no remainder along the deletion path: the entire gap between a first-order estimate and the exact deletion cost lies in the photometric loss (Section 4.2). Third, for the L_1 and L_2 losses this gap has closed forms. Adding it to the first-order term yields Corrected L_1 and Corrected L_2, which equal the exact single-Gaussian deletion cost in the ideal blending model and require one forward and one backward pass per view (Section 4.3).

4.1 Opacity gates and single-Gaussian deletion

Following the multiplicative-gate construction of SNIP (Lee et al., 2019), also adopted with learned masks in Gaussian compression pipelines (Lee et al., 2024; Liu et al., 2025), we attach a scalar gate

Table 2: Gaussian pruning scores described by scoring information, scored attributes, and aggregation.

Method	Source	Scoring information	Scored attributes	Aggregation
Plain-GS	plain	$J_v = \ell_{1,v}$	Opacity gate	Signed gradient sum
SNIP-GS	snip	$J_v = \ell_{1,v}$	Opacity gate	Sum of absolute gate derivatives
Fisher-GS	fisher	$J_v = \ell_{1,v}$	Opacity gate	Sum of squared gate derivatives
Jacov-GS	jacov	$J_v = \ell_{1,v}$	Opacity gate	Std. of normalized gradients
GraSP-GS	grasp	$J_v = \ell_{1,v}$	Opacity gate	Negative HVP (finite difference)
GradNorm-GS	grad-norm	$J_v = \ell_{1,v}$	Position, log-scale	Gradient-norm sums with block normalization
ZiCo-GS	zico	$J_v = \ell_{1,v}$	Position, log-scale	Sum of $\lvert \operatorname{mean}(g)\rvert/(\operatorname{std}(g)+\varepsilon)$
SynFlow-GS	synflow	$J_v = \mathbf{1}^\top R_v$	Opacity gate	Sum of absolute gate derivatives
PUP	Native	Rendered image elements	Position, log-scale	Log-determinant of Jacobian Gram
Zen-GS	zen-score	Perturbed-view difference	Opacity gate	Sum of absolute gate derivatives
EPE-GS	epe-nas	Perturbed-view difference	Opacity gate	Cluster-weighted absolute derivative sum
Usage-GS	flops	Rasterization hits	Activated opacity	Opacity-weighted hit sum
LightGaussian	Native	Rasterization hits	Opacity, volume	Opacity-weighted hit sum $\times$ volume factor
L2-GS	l2-norm	Primitive attributes	Activated opacity	Opacity magnitude
RAP	Native	Attributes and neighborhoods	Attribute features	Pretrained predictor
NASWOT-GS	nwot	Binary activity codes	Opacity, footprint	Kernel removal marginal
Corrected L_1	Derived	$J_v = \ell_{1,v}$	Opacity gate	Signed estimate + loss correction
Corrected L_2	Derived	$J_v = \lVert R_v - I_v\rVert_2^2$	Opacity gate	Signed estimate + loss correction

to activated opacity: $\widetilde{\alpha}_i = m_i\alpha_i$. The deletion path varies m_i from 1 to 0 while the underlying Gaussian parameters and all other gates remain fixed. The endpoint $m_i = 0$ removes the Gaussian from alpha blending. Zeroing position or stored log-scale instead moves its center to the origin or sets its activated scale to one; these zero endpoints do not represent deletion. We extend the notation $R_v(\mathbf{m})$ of Section 3.1 to continuous gates $\mathbf{m} \in [0,1]^N$; scores are evaluated at $\mathbf{m} = \mathbf{1}$. The gate derivative is

$$\left.\frac{\partial J}{\partial m_i}\right|_{\mathbf{m}=\mathbf{1}} = \alpha_i \frac{\partial J}{\partial \alpha_i}. \tag{3}$$

For a fixed pixel and color channel, front-to-back alpha blending composites the depth-ordered Gaussians by $C = \sum_k c_k a_k T_k$ with $T_k = \prod_{j<k}(1-a_j)$, where $a_k = \alpha_k \widehat{G}_k$ is the effective opacity (opacity times projected footprint) and T_k the transmittance. Scaling Gaussian i's opacity by its gate, $a_i \mapsto m_i a_i$, changes only its own emission term and the transmittance of the content behind it, both linearly in m_i; all other factors stay fixed. Hence:

Proposition 4.1 (Affine single-gate image)**.** *Fix the camera, footprints, depth order, the other Gaussians, and the background, with no opacity clipping, contribution thresholds, or early termination. Then, elementwise,*

$$C(m_i) = C(0) + m_i B_i, \qquad B_i = T_i\, a_i \left(c_i - C_i^{\mathrm{behind}}\right), \tag{4}$$

where C_i^{behind} is the color composited behind Gaussian i, T_i the prefix transmittance, and $C(0)$ is the image with Gaussian i deleted.

(Proof in Appendix B.) The deletion effect B_i thus combines three factors: the visibility reaching the Gaussian, its effective opacity, and its color contrast against the content behind. Opacity and footprint statistics capture chiefly the middle factor, which is why a large, frequently rendered Gaussian can still be cheap to delete. Since deletion is the endpoint $m_i = 0$, the deletion-induced image change equals minus the gate derivative: for view v, write $B_{v,i} = \partial R_v/\partial m_i|_{\mathbf{1}}$ and $D_{v,i} = R_v(\mathbf{1} - \mathbf{e}_i) - R_v(\mathbf{1})$; then

$$D_{v,i} = -B_{v,i}. \tag{5}$$

The deletion-induced image change of every Gaussian is thus determined by quantities of the ordinary blending pass (T_i, a_i, c_i, and C_i^{behind}), without re-rendering. Section 4.2 lifts this image identity to the loss.

4.2 FROM GATE DERIVATIVES TO FINITE DELETION COSTS

Section 4.1 characterized deletion at the level of the rendered image. A pruning score must instead estimate a scalar objective: the loss evaluated on that image, $J_v(\mathbf{m}) = \ell_v(R_v(\mathbf{m}), I_v)$. Its gate derivative is the image derivative propagated through the loss, and it is the loss's own nonlinearity along the gate that separates a first-order estimate from the exact deletion cost. Formally, define

$$d_{v,i}(J_v) = \frac{\partial J_v}{\partial m_i}\bigg|_{\mathbf{m}=\mathbf{1}}, \qquad \Delta J_{v,i} = J_v(\mathbf{1} - \mathbf{e}_i) - J_v(\mathbf{1}). \tag{6}$$

With $J_v = \ell_{1,v}$, summing $\Delta J_{v,i}$ over views recovers the reference deletion cost $\Delta\mathcal{E}_i$ of Section 3.1. Deleting Gaussian i takes the finite step $\Delta m_i = -1$, giving

$$\Delta J_{v,i} = -d_{v,i}(J_v) + \mathcal{R}_{v,i}, \tag{7}$$

where $-d_{v,i}(J_v)$ is the signed first-order estimate and the remainder $\mathcal{R}_{v,i}$ collects all higher-order terms. The first-order estimate is exact whenever the loss is affine along the deletion path.

Proposition 4.2 (Exactness condition)**.** *Suppose $C(m) = A + mB$ for $m \in [0,1]$. If $j(m) = J(C(m))$ is affine on this interval, then $J(C(0)) - J(C(1)) = -j'(1)$; in particular, an affine image objective satisfies this condition. Conversely, if the identity is required to hold for every realizable pair (A, B), then J must be affine in the image.*

(Proof in Appendix B.) Proposition 4.2 has two consequences. For the affine RGB-sum objective, the gate derivative is the exact deletion change, so SynFlow-GS aggregates exact deletion changes of the rendered intensity without correction. For any photometric loss, which is not affine, no first-order score can be exact: the remainder must be computed. Because the renderer is affine along the gate, this remainder is a function of the loss alone and admits closed forms, derived next.

4.3 LOSS-CORRECTED DELETION SCORES

Under the assumptions of Proposition 4.1, the image identity $D_{v,i} = -B_{v,i}$ of Section 4.1 turns the loss change into $\Delta J_{v,i} = \ell_v(R_v - B_{v,i}, I_v) - \ell_v(R_v, I_v)$ with $R_v = R_v(\mathbf{1})$, so the remainder $\mathcal{R}_{v,i} = \Delta J_{v,i} + d_{v,i}(J_v)$ in equation 7 has a closed form in $B_{v,i}$ for the L_1 and L_2 objectives. Adding it to the signed first-order term and aggregating across views gives the pruning score. Corrected L_1 uses the reference L_1 objective, while Corrected L_2 applies the same construction to the L_2 loss; both retain the sign of the loss change.

Corrected L_1. Let $R_{v,u}$ and $I_{v,u}$ denote image element u (a pixel and color channel) of the rendered and reference images, and let $r_{v,u} = R_{v,u} - I_{v,u}$; write $d_{v,i} = d_{v,i}(\ell_{1,v})$. Deletion changes the residual at element u to $r_{v,u} - B_{v,i,u}$, and the resulting loss remainder combined with the signed first-order estimate defines Corrected L_1:

$$s_i^{\mathrm{corr},L_1} = -\sum_v d_{v,i} + \underbrace{\sum_v \sum_u \left(|r_{v,u} - B_{v,i,u}| - |r_{v,u}| + \operatorname{sign}(r_{v,u}) B_{v,i,u}\right)}_{=:\,\overline{\mathcal{R}}_i \text{ (loss remainder)}}. \tag{8}$$

The remainder $\overline{\mathcal{R}}_i$ is nonnegative and vanishes wherever deletion leaves the residual sign unchanged.

Corrected L_2. The L_2 loss provides a smooth counterpart: under the ideal single-gate model, the loss is quadratic along the deletion path, and the same construction applied to the per-view L_2 loss $J_v^{(2)} = \sum_u (R_{v,u} - I_{v,u})^2$ gives

$$s_i^{\mathrm{corr},L_2} = \sum_v \left[-d_{v,i}^{(2)} + \|B_{v,i}\|_2^2\right], \qquad d_{v,i}^{(2)} = \partial J_v^{(2)}/\partial m_i|_{\mathbf{m}=\mathbf{1}}. \tag{9}$$

Here the correction $\|B_{v,i}\|_2^2$ is exactly half the diagonal loss curvature along the gate, and no higher-order term exists.

Corollary 4.3 (Exactness in the ideal model)**.** *Under the assumptions of Proposition 4.1, Corrected L_1 equals the signed single-Gaussian deletion cost $\Delta\mathcal{E}_i$ of the reference objective, and Corrected L_2 equals the exact signed change of the L_2 loss. (Proofs in Appendix B.)*

Why the correction dominates. In a well-fitted scene, residuals are small, while an important Gaussian contributes strongly to the pixels it covers: $|B_{v,i,u}| > |r_{v,u}|$. In this regime the first-order term fails exactly where it matters most. For L_1, wherever deletion flips the residual sign ($r_{v,u}$ and $B_{v,i,u}$ share a sign and $|B_{v,i,u}| > |r_{v,u}|$), the error changes by $|B_{v,i,u}| - 2|r_{v,u}|$, whereas the first-order term predicts $-|B_{v,i,u}|$; once $|B_{v,i,u}| > 2|r_{v,u}|$, the two have opposite signs. Wherever deletion leaves the residual sign unchanged, the first-order term is exact and $\overline{\mathcal{R}}_i$ receives no contribution. For L_2, the first-order term $-2r_{v,u}B_{v,i,u}$ vanishes as the residual shrinks, while the true cost approaches $B^2_{v,i,u}$. The closed-form remainder restores precisely this missing part, so the more accurate the reconstruction, the more the ranking depends on it. Section 5.3 confirms the prediction: without the remainder, signed first-order rankings are close to Random.

We retain the Gaussians with the largest corrected scores. Each score is computed for all Gaussians in one forward and one backward pass per view, whereas evaluating the same deletion cost by leave-one-out rendering would require one re-rendering per Gaussian and view. The scores are evaluated along the blending path recorded by the deployed rasterizer (implementation details in Appendix B).

5 Experiments

5.1 Experimental setup

Unified protocol. Every method is evaluated through one identical pipeline, and only the scoring stage differs, so differences in quality are attributable to the ranking alone. All scores start from the same scene-specific 30k-iteration 3DGS checkpoint and follow the two-round prune–refine schedule of PUP: remove 80% of the Gaussians, refine for 5k iterations, rescore, remove 50% of the survivors, and refine for another 5k, retaining 10% with densification disabled. Scores read only training views; PSNR, SSIM, and LPIPS are measured on the held-out views.

Scenes and baselines. We use all nine Mip-NeRF 360 scenes (Barron et al., 2022), train and truck from Tanks & Temples (Knapitsch et al., 2017), and drjohnson and playroom from Deep Blending (Hedman et al., 2018), with the standard 3DGS resolutions and test split (Appendix G). Baselines are the scoring stages of LightGaussian, PUP, and RAP (with its pretrained predictor), and Random averaged over five seeds; the unpruned reference is the 30k checkpoint. Scoring time covers both rounds on one NVIDIA L40S and excludes refinement (Appendix F.6); score definitions are given in Appendix E. Across two independent runs, the per-scene PSNR of the corrected scores differs by 0.004–0.005 dB on average and by at most 0.015 dB (Appendix F.5).

5.2 Main results

Corrected L_1 is the strongest pruning score in the evaluation (Table 3). It ranks first in eight of the nine dataset–metric comparisons; the ninth, Deep Blending PSNR, is led by Corrected L_2 (29.58 dB). Both corrected scores exceed PUP on all thirteen scenes, with mean per-scene gains of 0.58 and 0.54 dB, and Corrected L_1 also exceeds SNIP-GS and SynFlow-GS on every scene (Appendix F.2). With only 10% of the Gaussians, Corrected L_1 matches or exceeds the PSNR of the unpruned checkpoint on Tanks & Temples (23.42 vs. 23.39 dB) and Deep Blending (29.55 vs. 29.52 dB). The advantage grows with the pruning ratio: at 95% pruning, Corrected L_1 leads all nine comparisons against LightGaussian, PUP, SNIP-GS, and SynFlow-GS, and its Mip-NeRF 360 margin over PUP widens from 0.59 to 0.89 dB, with the same trend on the other two datasets (Appendix F.4). Visually, Corrected L_1 preserves background structure and object boundaries that PUP loses (Figure 1 in Appendix F.3).

Exact deletion costs are cheap. Corrected L_1 and Corrected L_2 score in 8.58 and 8.47 s, about one-third of PUP's 25.22 s and below RAP's 9.67 s; the closed-form remainder adds only about 10% to the cost of a plain gate-gradient score such as SNIP-GS (7.79 s; per-round times in Appendix F.6).

Deletion alignment predicts which proxies transfer. The seven leading adapted proxies (the shaded upper block of Table 3) exceed all three published scores in mean PSNR on every dataset; on Mip-NeRF 360, SynFlow-GS improves over PUP by 0.47 dB. Six of the seven differentiate the opacity gate, the coordinate that represents deletion. NASWOT-GS, L2-GS, and ZiCo-GS, which rank kernel removal marginals, opacity magnitude, and cross-view gradient consistency, fall below Random on every dataset. GraSP-GS differentiates the gate but ranks a change of gradient energy

Table 3: Reconstruction quality at 90% pruning and scoring time. Shaded cells improve on the best published 3DGS score (LightGaussian, PUP, or RAP) in that column. Bold and underline mark the best and second-best values among pruned methods; ties at the displayed precision share the marking. Adapted proxies are sorted by Mip-NeRF 360 PSNR. The unpruned reference receives no additional refinement and is excluded from ranking.

	Mip-NeRF 360			Tanks & Temples			Deep Blending			Scoring
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	Time (s)↓
References										
3DGS (unpruned)	27.35	0.8142	0.2166	23.39	0.8416	0.1828	29.52	0.9030	0.2422	—
Random (5 seeds)	25.78	0.7449	0.3294	22.42	0.7871	0.2768	28.99	0.8955	0.2764	—
Published 3DGS scores										
LightGaussian	26.22	0.7649	0.2976	22.89	0.7946	0.2634	29.12	0.8900	0.2826	2.81
PUP	26.57	0.7856	0.2678	22.45	0.7987	0.2479	29.40	0.9025	0.2557	25.22
RAP	26.36	0.7659	0.3063	22.88	0.7896	0.2769	29.03	0.8886	0.2929	9.67
Adapted zero-cost proxies (this work)										
SynFlow-GS	27.04	0.7967	0.2606	<u>23.38</u>	0.8199	0.2313	29.53	0.9036	0.2554	7.71
Zen-GS	27.02	<u>0.7987</u>	<u>0.2601</u>	23.34	<u>0.8201</u>	<u>0.2311</u>	29.52	0.9040	0.2548	15.78
SNIP-GS	27.01	0.7953	0.2627	23.34	0.8175	0.2325	29.50	0.9025	0.2582	7.79
GradNorm-GS	26.98	0.7949	0.2654	23.13	0.8083	0.2434	29.54	0.9038	0.2577	7.84
Jacov-GS	26.98	0.7918	0.2685	23.33	0.8163	0.2372	29.49	0.9023	0.2586	7.78
EPE-GS	26.97	0.7976	0.2621	23.27	0.8165	0.2355	29.54	0.9041	0.2546	21.85
Fisher-GS	26.96	0.7899	0.2718	23.36	0.8171	0.2353	29.49	0.9020	0.2591	7.74
GraSP-GS	26.40	0.7704	0.2927	22.68	0.7940	0.2608	29.35	0.8987	0.2667	15.20
Usage-GS	26.08	0.7680	0.2949	22.69	0.7877	0.2695	29.09	0.8896	0.2840	2.73
Plain-GS	25.80	0.7657	0.3021	22.11	0.7879	0.2697	28.83	0.8941	0.2762	7.72
ZiCo-GS	25.48	0.7463	0.3214	22.35	0.7842	0.2758	28.92	0.8931	0.2838	7.92
L2-GS	25.11	0.7072	0.3658	22.33	0.7756	0.2928	28.61	0.8848	0.2940	—
NASWOT-GS	24.19	0.7339	0.3328	20.74	0.7531	0.3026	28.76	0.8840	0.2937	3.78
Corrected deletion scores (this work)										
Corrected L_1	**27.16**	**0.8021**	**0.2535**	**23.42**	**0.8230**	**0.2288**	<u>29.55</u>	**0.9046**	**0.2532**	8.58
Corrected L_2	<u>27.11</u>	0.7950	0.2664	23.36	0.8177	0.2396	**29.58**	<u>0.9045</u>	<u>0.2545</u>	8.47

Time is the two-round scoring total averaged over thirteen scenes; — indicates unavailable or inapplicable timing.

rather than of the loss (Appendix C), and trails PUP on Mip-NeRF 360 and Deep Blending. A score performs well when both its coordinate and its aggregation point at deletion; Sections 5.3 and 5.4 isolate each factor.

5.3 The loss correction

Table 4 fixes the opacity gate and the loss and varies only how the gate derivatives $d_{v,i}$ are turned into a score. The table traces the path from first-order information to the exact deletion cost.

The signed first-order estimate carries almost no ranking information. On Mip-NeRF 360, the first-order deletion estimate $-\sum_v d_{v,i}$ reaches 25.66 dB and its unnegated counterpart Plain-GS 25.80 dB, against 25.78 dB for Random; on every dataset both stay within 0.35 dB of Random, as Section 4.3 predicts for a well-fitted scene.

Magnitude aggregation recovers most of the gap. SNIP-GS and Fisher-GS discard the unreliable sign and bound the magnitude of the linearized deletion change (Appendix C), lifting Mip-NeRF 360 PSNR to 27.01 and 26.96 dB.

The exact remainder completes it. Adding $\overline{\mathcal{R}}_i$ to the same signed estimate lifts Mip-NeRF 360 PSNR from 25.66 to 27.16 dB, and the L_2 remainder from 25.51 to 27.11 dB; Corrected L_1 also exceeds SNIP-GS by 0.15 dB. On all three datasets, each corrected score exceeds its first-order control, and Corrected L_1 exceeds SNIP-GS and Fisher-GS, in all nine dataset–metric comparisons. Since each pair shares the gate, loss, and normalization, the gain is attributable to the remainder alone (Appendix E.2).

Table 4: From first-order gate information to exact deletion cost. All rows use the opacity gate; the L_1 block fixes the L_1 loss and varies only the aggregation of the gate derivatives $d_{v,i}$. Each corrected score shares the loss, normalization, and gate of its signed first-order control; $\overline{\mathcal{R}}_i$ is the closed-form L_1 loss remainder of equation 8. Bold marks the best value within the L_1 block.

		Mip-NeRF 360			Tanks & Temples			Deep Blending		
Score	Aggregation	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Random	—	25.78	0.7449	0.3294	22.42	0.7871	0.2768	28.99	0.8955	0.2764
Plain-GS	$\sum_v d_{v,i}$	25.80	0.7657	0.3021	22.11	0.7879	0.2697	28.83	0.8941	0.2762
Signed L_1 first-order	$-\sum_v d_{v,i}$	25.66	0.7260	0.3389	22.57	0.7807	0.2805	29.08	0.8933	0.2777
Fisher-GS	$\sum_v d_{v,i}^2$	26.96	0.7899	0.2718	23.36	0.8171	0.2353	29.49	0.9020	0.2591
SNIP-GS	$\sum_v \lvert d_{v,i}\rvert$	27.01	0.7953	0.2627	23.34	0.8175	0.2325	29.50	0.9025	0.2582
Corrected L_1	$-\sum_v d_{v,i} + \overline{\mathcal{R}}_i$	**27.16**	**0.8021**	**0.2535**	**23.42**	**0.8230**	**0.2288**	**29.55**	**0.9046**	**0.2532**
Signed L_2 first-order	$-\sum_v d_{v,i}^{(2)}$	25.51	0.7195	0.3448	22.35	0.7754	0.2872	28.90	0.8908	0.2801
Corrected L_2	$-\sum_v d_{v,i}^{(2)} + \sum_v \lVert B_{v,i}\rVert_2^2$	27.11	0.7950	0.2664	23.36	0.8177	0.2396	29.58	0.9045	0.2545

Table 5: PSNR when the differentiated coordinate is changed from the spatial parameters $(\boldsymbol{\mu}_i, \boldsymbol{\rho}_i)$ (position and stored log-scale) to the opacity gate m_i, with objective and aggregation fixed within each row. For PUP, the opacity-gate column is PUP-α, which applies PUP's log-Gram scalarization to m_i (Appendix D).

	Mip-NeRF 360		Tanks & Temples		Deep Blending	
Method	Spatial	Opacity gate	Spatial	Opacity gate	Spatial	Opacity gate
SNIP-GS	26.73	**27.01**	22.99	**23.34**	29.43	**29.50**
SynFlow-GS	26.78	**27.04**	23.11	**23.38**	29.50	**29.53**
PUP / PUP-α	26.57	**26.83**	22.45	**23.00**	29.40	**29.48**

5.4 THE DELETION COORDINATE

Table 5 keeps each method's objective and aggregation fixed and changes only the differentiated coordinate, from stored position and log-scale to the opacity gate. **The opacity gate improves PSNR in all nine method–dataset comparisons.** On Mip-NeRF 360 it gains 0.28 dB for SNIP-GS and 0.26 dB for SynFlow-GS, positive on eight and nine of the nine scenes, respectively. The effect extends to PUP itself: PUP-α, which applies PUP's own curvature scalarization to the opacity gate, gains 0.26 dB over spatial PUP. The coordinate must represent deletion, not merely involve opacity. Differentiating the opacity logit multiplies the gate derivative by $1 - \alpha_i$ and suppresses exactly the opaque Gaussians whose deletion is most costly; the resulting GradNorm-GS (opacity logit) falls to 25.89 dB on Mip-NeRF 360, against 27.01 dB for the rank-equivalent gate score SNIP-GS (Appendix F.1).

Once the coordinate is the opacity gate, reference images become optional. SynFlow-GS differentiates the rendered RGB sum, for which the gate derivative is the exact deletion change (Proposition 4.2); without reading any reference image, it matches SNIP-GS on Mip-NeRF 360 (27.04 vs. 27.01 dB) and on the other two datasets.

6 CONCLUSION

Prune by Proxy treats every Gaussian pruning score as a proxy of the single-Gaussian deletion cost, which makes transferred zero-cost proxies and published 3DGS scores directly comparable: the more precisely a score points at deletion, the better it prunes. In 3DGS this alignment can be made exact. Because alpha blending is affine in a single opacity gate, the deletion cost of every Gaussian equals a gradient plus a closed-form loss remainder, computed in one forward and one backward pass per view; the resulting Corrected L_1 exceeds PUP on all thirteen scenes by 0.58 dB on average at one-third of its scoring time. Since blending remains multilinear in the gates, the same construction offers a direct route to exact scores for jointly deleted subsets.

AI USE STATEMENT

Large language models assisted with drafting, restructuring, LaTeX editing, and checks of mathematical and implementation consistency. No new experimental measurements were generated by a language model. The authors are responsible for verifying the claims, measurements, references, and final submission.

ETHICS STATEMENT

This work evaluates post-hoc pruning using public scene-reconstruction benchmarks and existing reconstructed scenes. Any use or redistribution of those scenes remains subject to the source datasets' usage conditions.

REPRODUCIBILITY STATEMENT

The scoring interface and constructions are described in Sections 3 and 4; the common experimental protocol is in Section 5.1 and Appendix G. The appendix provides proofs, score implementation details, extended results, and repeated-run diagnostics.

REFERENCES

Mohamed S. Abdelfattah, Abhinav Mehrotra, Łukasz Dudziak, and Nicholas D. Lane. Zero-cost proxies for lightweight NAS. In *International Conference on Learning Representations (ICLR)*, 2021.

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)*, 2022.

Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. HAC: Hash-grid assisted context for 3D gaussian splatting compression. In *European Conference on Computer Vision (ECCV)*, 2024.

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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

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

Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. EAGLES: Efficient accelerated 3D gaussians with lightweight EncodingS. In *European Conference on Computer Vision (ECCV)*, 2024.

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)*, 2025.

Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics*, 2018.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 2023.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 2017.

Arjun Krishnakumar, Colin White, Arber Zela, Renbo Tu, Mahmoud Safari, and Frank Hutter. NAS-Bench-Suite-Zero: Accelerating research on zero cost proxies. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2022.

Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems (NIPS)*, 1989.

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)*, 2024.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip H. S. Torr. SNIP: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations (ICLR)*, 2019.

Guihong Li, Yuedong Yang, Kartikeya Bhardwaj, and Radu Marculescu. ZiCo: Zero-shot NAS via inverse coefficient of variation on gradients. In *International Conference on Learning Representations (ICLR)*, 2023.

Ming Lin, Pichao Wang, Zhenhong Sun, Hesen Chen, Xiuyu Sun, Qi Qian, Hao Li, and Rong Jin. Zen-NAS: A zero-shot NAS for high-performance image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Xiang Liu, Yimin Zhou, Jinxiang Wang, Yujun Huang, Shuzhao Xie, Shiyu Qin, Mingyao Hong, Jiawei Li, Yaowei Wang, Zhi Wang, Shu-Tao Xia, and Bin Chen. Splatwizard: A benchmark toolkit for 3D gaussian splatting compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, 2026.

Yifei Liu, Zhihang Zhong, Yifan Zhan, Sheng Xu, and Xiao Sun. MaskGaussian: Adaptive 3D gaussian representation from probabilistic masks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Vasco Lopes, Saeid Alirezazadeh, and Luís A. Alexandre. EPE-NAS: Efficient performance estimation without training for neural architecture search. In *International Conference on Artificial Neural Networks (ICANN)*, 2021.

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)*, 2024.

Joe Mellor, Jack Turner, Amos Storkey, and Elliot J. Crowley. Neural architecture search without training. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

Simon Niedermayr, Josef Stumpfegger, and Rüdiger Westermann. Compressed 3D gaussian splatting for accelerated novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

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 *International Conference on 3D Vision (3DV)*, 2025.

Hidenori Tanaka, Daniel Kunin, Daniel L. K. Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Lucas Theis, Iryna Korshunova, Alykhan Tejani, and Ferenc Huszár. Faster gaze prediction with dense networks and Fisher pruning. *arXiv preprint arXiv:1801.05787*, 2018. URL `https://arxiv.org/abs/1801.05787`.

Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations (ICLR)*, 2020.

Haoran Wang, Guoxi Huang, Fan Zhang, David Bull, and Nantheera Anantrasirichai. Prune wisely, reconstruct sharply: Compact 3D gaussian splatting via adaptive pruning and difference-of-gaussian primitives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Kaifa Yang, Qi Yang, Yiling Xu, and Zhu Li. RAP: Fast feedforward rendering-free attribute-guided primitive importance score prediction for efficient 3D gaussian splatting processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

A NOTATION

Table 6 collects the main symbols. Subscripts v and u index views and image elements; symbols used only within a single derivation are defined where they appear.

Table 6: Notation.

Symbol	Meaning
$\mathcal{G} = \{g_i\}_{i=1}^{N}$	Trained scene of N Gaussians
$\boldsymbol{\mu}_i, \mathbf{s}_i, \mathbf{r}_i, \mathbf{c}_i, \alpha_i$	Position, activated scale, rotation, color, activated opacity
$\boldsymbol{\rho}_i$, o_i	Stored log-scale $\log \mathbf{s}_i$; stored opacity logit
V, ϕ_v, I_v	Number of training views; camera and reference image
$R_v(\mathbf{m})$, R_v	Rendered image under mask or gates $\mathbf{m}$; $R_v = R_v(\mathbf{1})$
$\mathbf{m}$, $\mathbf{e}_i$	Mask or gates; ith unit vector ($\mathbf{1} - \mathbf{e}_i$ deletes g_i)
J, J_v, ℓ_v	Scalar objective; its per-view term; per-view image loss
ΔJ_i, $\Delta J_{v,i}$	Single-Gaussian deletion cost; per-view deletion cost
$\ell_{1,v}$, $\mathcal{E}$	Per-view L_1 error; reference objective
$\Delta\mathcal{E}_i$	Reference deletion cost; every score is treated as a proxy of it
$J_v^{(2)}$	Per-view L_2 loss
m_i, $\widetilde{\alpha}_i$	Opacity gate; gated opacity $m_i\alpha_i$
$\widehat{G}_k$, a_k, T_k	Projected footprint, effective opacity, transmittance on a ray
C, C_i^{behind}	Composited color; color composited behind g_i
B_i, $B_{v,i}$	Gate derivative of the color on a ray and of the image R_v
$D_{v,i}$	Deletion-induced image change, $D_{v,i} = -B_{v,i}$
$d_{v,i}(J_v)$	Gate derivative of J_v at $\mathbf{1}$; $d_{v,i}$, $d_{v,i}^{(2)}$ for L_1, L_2
$\mathcal{R}_{v,i}$, $\overline{\mathcal{R}}_i$	Per-view remainder; view-summed L_1 remainder
$r_{v,u}$	Residual $R_{v,u} - I_{v,u}$
$s_i^{(\cdot)}$, K	Score of g_i; number of retained Gaussians

B DELETION IDENTITIES AND LOSS REMAINDERS

Chain rule and deletion. For a gate on an attribute vector, $\widetilde{\mathbf{a}}_i = m_i\mathbf{a}_i$, the scalar gate derivative is $\partial J/\partial m_i = \mathbf{a}_i^\top \nabla_{\widetilde{\mathbf{a}}_i} J$; independent componentwise gates give elementwise products, which the spatial parameter-product controls sum in absolute value. For activated opacity, the block has one component. Setting it to zero sets its color contribution to zero and passes the incoming transmittance through without attenuation, exactly as when the primitive is removed from the depth-ordered blending sequence.

Spatial coordinates. Position and scale perturbations alter projection and visibility and have no endpoint that deletes the Gaussian. Zeroing the stored log-scale $\boldsymbol{\rho}_i = \log \mathbf{s}_i$ produces unit activated scale, and even zero activated scale leaves a footprint because the rasterizer imposes a screen-space covariance floor. Spatial sensitivities also depend on scene units and parameterization, whereas the opacity gate is dimensionless.

Proof of Proposition 4.1. Fix a pixel and a color channel, index the Gaussians on the ray by depth order $k = 1, \ldots, K$, and let i also denote the position of Gaussian i in this order. Write $a_k = \alpha_k \widehat{G}_k$ for the effective opacities, $T_k = \prod_{j<k}(1 - a_j)$ for the transmittances, and C^{bg} for the background color (zero in the expression of Section 4.1). The gate replaces a_i by $m_i a_i$ and leaves every other

a_k unchanged, so the gated transmittances are

$$\widetilde{T}_k = T_k \quad (k \le i), \qquad \widetilde{T}_k = T_i\,(1 - m_i a_i) \prod_{i<j<k} (1 - a_j) \quad (k > i),$$

with $\widetilde{T}_{K+1}$ the transmittance reaching the background. Splitting the composite at position i gives

$$C(m_i) = \underbrace{\sum_{k<i} c_k a_k T_k}_{=:\,P_i} + c_i\, m_i a_i\, T_i + \sum_{k>i} c_k a_k \widetilde{T}_k + \widetilde{T}_{K+1} C^{\mathrm{bg}}.$$

Every term behind Gaussian i carries the common factor $T_i(1 - m_i a_i)$, so the last two terms equal $T_i(1 - m_i a_i)\, C_i^{\mathrm{behind}}$ with

$$C_i^{\mathrm{behind}} = \sum_{k>i} c_k a_k \prod_{i<j<k} (1 - a_j) + \prod_{j>i} (1 - a_j)\, C^{\mathrm{bg}},$$

the color composited strictly behind Gaussian i with starting transmittance 1. Hence

$$C(m_i) = P_i + T_i\big[m_i a_i c_i + (1 - m_i a_i) C_i^{\mathrm{behind}}\big] = \underbrace{P_i + T_i C_i^{\mathrm{behind}}}_{C(0)} + m_i \underbrace{T_i a_i \big(c_i - C_i^{\mathrm{behind}}\big)}_{B_i}.$$

The quantities P_i, T_i, a_i, c_i, and C_i^{behind} involve only the fixed footprints, the depth order, and the other Gaussians, and are therefore independent of m_i. Thus C is affine in m_i, and its slope B_i equals $\partial C/\partial m_i$ at every m_i, in particular at $m_i = 1$. Applying the identity to every pixel and channel of view v gives $R_v(\mathbf{1} - \mathbf{e}_i) - R_v(\mathbf{1}) = C(0) - C(1) = -B_{v,i}$, which is equation 5. The assumptions of the proposition enter only through the exact replacement $a_i \mapsto m_i a_i$ and the complete, fixed ray list; no assumption is made on the content or occlusion behind Gaussian i.

Proof of Proposition 4.2. Let $C(m) = A + mB$ and $j(m) = J(C(m))$. The exact change is $J(C(0)) - J(C(1)) = j(0) - j(1)$, and the first-order estimate is $-j'(1) = -\nabla J(A + B)^\top B$.

Sufficiency. If $j(m) = a + bm$ on $[0, 1]$, then $j(0) - j(1) = -b = -j'(1)$. If J is affine in the image, $J(x) = \mathbf{w}^\top x + w_0$, then $j(m) = \mathbf{w}^\top A + w_0 + m\,\mathbf{w}^\top B$ is affine for every pair (A, B), so the identity holds for every Gaussian.

Necessity. Let U be an open convex set of images on which J is differentiable, and call a pair (A, B) realizable if both endpoints $C(0) = A$ and $C(1) = A + B$ lie in U. For any $x, y \in U$, the pair $A = y$, $B = x - y$ is realizable, and the identity reads

$$J(y) = J(x) + \nabla J(x)^\top (y - x).$$

Fixing $x = x_0$ shows $J(y) = J(x_0) + \nabla J(x_0)^\top (y - x_0)$ for all $y \in U$, so J coincides with its tangent plane at x_0 and is affine on U. In particular, the L_1 and L_2 losses are not affine, and a first-order score of either loss has a nonzero remainder for some Gaussians.

Proof of Corollary 4.3. We show $s_i^{\mathrm{corr},L_1} = \Delta\mathcal{E}_i$, with $\Delta\mathcal{E}_i$ the reference deletion cost of Section 3.1, and that s_i^{corr,L_2} equals the exact signed change of the L_2 loss.

Setup. Fix a view v (suppressed below) and an image element u. By Proposition 4.1, the rendered color is affine in the gate, $C_u(m_i) = C_u(0) + m_i B_{i,u}$, where $B_{i,u} = \partial C_u/\partial m_i|_{m_i=1}$ is the gate derivative of the image at element u. With $r_u = C_u(1) - I_u$ the residual at the current scene, the residual along the deletion path is

$$C_u(m_i) - I_u = r_u + (m_i - 1) B_{i,u}. \tag{10}$$

Write $j(m) = J(C(m))$ for the objective along this path. Then $d_{v,i} = j'(1)$ is the signed first-order estimate, $\Delta J = j(0) - j(1)$ is the exact per-view deletion change, and the remainder is $\mathcal{R} = \Delta J + j'(1)$.

L_1. Here $j(m) = \sum_u |r_u + (m - 1)B_u|$, the per-view L_1 objective. With the implemented subgradient $\mathrm{sign}(0) = 0$, differentiating elementwise gives

$$j'(1) = \sum_u \mathrm{sign}(r_u) B_u. \tag{11}$$

The exact change is

$$\Delta J = j(0) - j(1) = \sum_u \big(|r_u - B_u| - |r_u|\big), \tag{12}$$

so the remainder is

$$\mathcal{R} = \sum_u \big(|r_u - B_u| - |r_u| + \operatorname{sign}(r_u) B_u\big). \tag{13}$$

Each element contributes a nonnegative term: absolute value is convex, and $\operatorname{sign}(r)$ is a subgradient of $|\cdot|$ at r, so $|r_u - B_u| \geq |r_u| + \operatorname{sign}(r_u)\big((r_u - B_u) - r_u\big) = |r_u| - \operatorname{sign}(r_u) B_u$. Elementwise, the term is $2|r_u - B_u|$ when r_u and $r_u - B_u$ have strictly opposite signs, $|B_u|$ when $r_u = 0$, and 0 otherwise. Hence the per-view identity is $\Delta\ell_{1,v,i} = -d_{v,i} + \sum_u \big(|r_u - B_{v,i,u}| - |r_u| + \operatorname{sign}(r_u) B_{v,i,u}\big)$, and summing over views gives $s_i^{\mathrm{corr},L_1} = \sum_v \Delta\ell_{1,v,i} = \Delta\mathcal{E}_i$.

L_2. Here $j(m) = \sum_u (r_u + (m-1)B_u)^2$, exactly quadratic in m, with

$$j'(1) = 2\sum_u r_u B_u, \qquad j''(m) = 2\sum_u B_u^2. \tag{14}$$

The exact change is

$$\Delta J = \sum_u \big[(r_u - B_u)^2 - r_u^2\big] = -2\sum_u r_u B_u + \sum_u B_u^2 = -j'(1) + \|B\|_2^2. \tag{15}$$

The correction $\|B_{v,i}\|_2^2$ therefore equals $j''/2$, half the diagonal loss Hessian along the ideal opacity gate, and there is no higher-order remainder. Hence $s_i^{\mathrm{corr},L_2} = \sum_v \Delta J_{v,i}^{(2)}$, the exact signed change of the per-view L_2 loss summed over views.

Joint deletion. For a deleted set S, the same argument applied gate by gate shows that the ideal rendered image is multilinear in $(m_i)_{i\in S}$, with degree at most $|S|$. Pairwise and higher interaction terms between overlapping Gaussians are therefore polynomial with explicit coefficients, which makes exact subset scoring a direct extension of the single-Gaussian construction.

Deployed rasterizer. Proposition 4.1 describes the unclipped blending model. The deployed forward kernel additionally clips effective opacity at $a = \min(0.99, \alpha\widehat{G})$, skips contributions below $1/255$, and terminates a ray once transmittance falls below 10^{-4}. The opacities and colors of the other Gaussians do not depend on m_i, so the deletion identity $D_{v,i} = -B_{v,i}$ continues to hold except where deleting Gaussian i moves a quantity across one of these thresholds: a saturated effective opacity, or a suffix hidden by early termination that deletion would reveal. The corrected scores are evaluated with the transmittances and colors recorded by the deployed kernel, without a separate rendering for each Gaussian.

C Mathematical links to the common deletion objective

This appendix places each score relative to the reference deletion cost $\Delta\mathcal{E}_i$ of Section 3.1. The relations explain the ordering in Section 5: scores that bound or reproduce a deletion change transfer well, and scores that measure a different quantity do not. All image norms sum over pixels and color channels. By Corollary 4.3, $\Delta\mathcal{E}_i = -\sum_v d_{v,i} + \overline{\mathcal{R}}_i$ with $d_{v,i} = \partial\ell_{1,v}/\partial m_i|_{\mathbf{1}}$ and $\overline{\mathcal{R}}_i \geq 0$ from equation 8; dropping the remainder gives the signed linearized estimate $\widehat{\Delta\mathcal{E}}_i = -\sum_v d_{v,i}$ used below.

SNIP-GS and Fisher-GS: bounds on a linearized magnitude. For $s_i^{\mathrm{SNIP}} = \sum_v |d_{v,i}|$ and $s_i^{\mathrm{Fisher}} = \sum_v d_{v,i}^2$, triangle and Cauchy–Schwarz inequalities yield

$$\left|\widehat{\Delta\mathcal{E}}_i\right| \leq s_i^{\mathrm{SNIP}} \leq \sqrt{V\, s_i^{\mathrm{Fisher}}}. \tag{16}$$

The factor V and the square root preserve the Fisher-GS ranking. Both scores thus bound the magnitude of the linearized deletion change while discarding its sign, which Section 5.3 shows to be unreliable. Under the ideal model,

$$|\Delta\mathcal{E}_i| \leq s_i^{\mathrm{SNIP}} + \overline{\mathcal{R}}_i \leq \sqrt{V\, s_i^{\mathrm{Fisher}}} + \overline{\mathcal{R}}_i,$$

so the part of the deletion cost that both scores miss is exactly the remainder $\overline{\mathcal{R}}_i$ that Corrected L_1 adds.

GradNorm-GS: gate versus logit. For a scalar opacity gate, the gradient norm equals its absolute value, so gate GradNorm and SNIP-GS yield the same ranking under single-block normalization. With the stored logit o_i and $\alpha_i = \sigma(o_i)$, $\partial J/\partial o_i = (1-\alpha_i)\,\partial J/\partial m_i$; the logit score is SNIP-GS reweighted by $1-\alpha_i$, which demotes the most opaque Gaussians.

SynFlow-GS: exact deletion change of intensity. For $J_v^{\mathrm{sum}} = \mathbf{1}^\top R_v$, ideal single-gate affinity gives $\Delta J_{v,i}^{\mathrm{sum}} = -\mathbf{1}^\top B_{v,i}$ and

$$|\Delta J_{v,i}^{\mathrm{sum}}| = |\mathbf{1}^\top B_{v,i}| \le \|B_{v,i}\|_1. \tag{17}$$

SynFlow-GS accumulates the left side over views: the exact deletion change of rendered intensity, obtained without a reference image. It measures the net image change of deletion, whereas the corrected scores measure its effect on the photometric error.

Opacity-gate Grams and the L_2 correction. For the scalar opacity gate, the image-Jacobian Gram of PUP reduces to $\sum_{v,u} B_{v,i,u}^2$, which is exactly the correction of Corrected L_2 in equation 15. Corrected L_2 therefore combines the gate version of PUP's curvature statistic with the signed first-order term, and the sum equals the exact L_2 deletion cost. Spatial PUP instead takes a log-determinant of a 6×6 position/scale Gram, which measures geometric perturbation rather than deletion.

GraSP-GS and gradient statistics. On a twice-differentiable branch, let $g_v = \nabla_{\mathbf{m}} J_v$ and $H_v = \nabla_{\mathbf{m}}^2 J_v$. For gradient energy $G_v(\mathbf{m}) = \frac{1}{2}\|g_v\|_2^2$,

$$\nabla_{\mathbf{m}} G_v = H_v g_v.$$

The implemented negative Hessian–vector product is therefore the first-order deletion change of gradient energy, not of the loss. Jacov-GS and ZiCo-GS summarize cross-view variation or consistency of gradients, which likewise are not deletion changes.

Attribute and contribution statistics. Under ideal alpha blending, $B_{v,i,u} = T_{v,i,u}\alpha_i \widehat{G}_{v,i,u}(c_{v,i,u} - C_{v,i,u}^{\mathrm{behind}})$. If all color contrasts are bounded in magnitude by M, then

$$\|B_{v,i}\|_1 \le M\alpha_i \sum_u \widehat{G}_{v,i,u}.$$

Opacity and footprint statistics (Usage-GS, LightGaussian, L2-GS) retain only this coarse bound and discard the visibility and contrast factors of the deletion effect.

Kernel marginals. For a retained set U with fixed feature vectors $\mathbf{f}_j$ and $\lambda > 0$, define $A_U = \lambda I + \sum_{j\in U} \mathbf{f}_j \mathbf{f}_j^\top$. The matrix determinant lemma gives the exact fixed-feature information loss

$$\log\det A_U - \log\det A_{U\setminus\{i\}} = -\log(1 - \mathbf{f}_i^\top A_U^{-1}\mathbf{f}_i). \tag{18}$$

NASWOT-GS ranks this marginal for binary activity codes: a diversity objective over features, unrelated to the photometric deletion cost.

D PUP AND CURVATURE INTERPRETATIONS

Individual image-output derivatives. For image element u at the scoring resolution, let $\mathbf{j}_{v,u,i} = \nabla_{z_i} R_{v,u}$. PUP forms the local Gram

$$F_i = \sum_{v,u} \mathbf{j}_{v,u,i}\mathbf{j}_{v,u,i}^\top. \tag{19}$$

Spatial PUP uses $z_i = (\boldsymbol{\mu}_i, \boldsymbol{\rho}_i)$ and a 6×6 block; PUP-α uses $z_i = m_i$. Both accumulate local outer products of individual image-element derivatives before scalarization, which distinguishes them from Fisher-GS's squared loss derivatives.

Numerical accumulation. The CUDA implementation reduces image dimensions by the scoring factor q, rounding upward, and accumulates per-pixel/channel Jacobian outer products over position and log-scale in the 21 independent entries of a symmetric 6×6 block, implementing the local Gram in equation 19. The reported 4×4 and 1×1 variants denote $q = 4$ and $q = 1$.

The deployed spatial score is

$$s_i^{\mathrm{PUP}} = \sum_{k=1}^{6} \log \sigma_k(F_i). \tag{20}$$

A singular spatial block yields $-\infty$. PUP-α accumulates the squared local opacity gate derivatives and uses $\log \max(F_i, 10^{-12})$, with the same scoring-resolution factor as spatial PUP.

Relation to second-order pruning. Optimal Brain Damage (LeCun et al., 1989) approximates the loss change with a diagonal Hessian near a stationary point. For squared residual loss,

$$\nabla_\theta^2 \tfrac{1}{2} \|R - I\|_2^2 = J_R^\top J_R + \sum_u (R_u - I_u) \nabla_\theta^2 R_u. \tag{21}$$

In spatial coordinates the second term is generally nonzero and indefinite, and PUP's Gauss–Newton block drops it together with all cross-Gaussian interactions. Along the opacity gate, the ideal image is affine, so $\nabla_{m_i}^2 R_u = 0$: the second term vanishes identically and the Gauss–Newton curvature $\|B_{v,i}\|_2^2$ is the exact curvature. Corrected L_2 uses it in closed form, without the double backward pass that the rasterizer does not provide.

E Proxy sources and score definitions

E.1 Source catalog

The adapted proxies belong to the following source types of NAS-Bench-Suite-Zero's Table 1 (Krishnakumar et al., 2022):

- Pruning-at-initialization: `grad-norm`, `snip`, `grasp`, `fisher`, and `synflow`.
- Jacobian: `jacov`, `epe-nas`, and `nwot`.
- Baseline: `flops`, `l2-norm`, and `plain`.
- Piecewise linearity: `zen-score`.

These categories are those of the source NAS catalog. Table 2 describes the actual 3DGS scoring information, scored attributes, and aggregation. ZiCo-GS adapts a later proxy outside this catalog.

E.2 Scoring constructions

Unless otherwise stated, scores accumulate over training cameras and retain larger values. SNIP-GS and SynFlow-GS use opacity gates; spatial variants are labeled explicitly. PUP uses spatial coordinates and scoring-resolution factor four unless marked otherwise. L2-GS ranks activated opacity and does not use a squared photometric loss. Usage-GS ranks opacity-weighted rasterization hits, not a measured FLOP count.

SNIP-GS and SynFlow-GS. For $d_{v,i}(J_v) = \partial J_v / \partial m_i|_{\mathbf{1}}$, SNIP-GS and SynFlow-GS use the same opacity gate and sum of absolute gate derivatives, but differentiate different objectives. For SNIP-GS, write $d_{v,i} = d_{v,i}(\ell_{1,v})$, using $\mathrm{sign}(0) = 0$ at zero residuals. The score is

$$s_i^{\mathrm{SNIP}} = \sum_v |d_{v,i}|, \qquad J_v = \ell_{1,v} = \|R_v - I_v\|_1. \tag{22}$$

SynFlow-GS instead differentiates the rendered RGB sum:

$$s_i^{\mathrm{SynFlow}} = \sum_v |d_{v,i}(J_v)|, \qquad J_v = \mathbf{1}^\top R_v. \tag{23}$$

SynFlow-GS and the source method. SynFlow (Tanaka et al., 2020) scores a data-free output sum through absolute network weights. SynFlow-GS keeps the data-free output-sum objective and differentiates the actual renderer; because opacity controls both emission and transmittance, its justification comes from the single-gate identity of Proposition 4.2 rather than from SynFlow's conservation argument.

GradNorm-GS and spatial products. For stored blocks $b \in \{\boldsymbol{\mu}, \boldsymbol{\rho}\}$ with $\boldsymbol{\rho} = \log \mathbf{s}$, GradNorm-GS normalizes each block after accumulating its per-view gradient norms:

$$t_{i,b} = \sum_v \|\nabla_{b_i} \ell_{1,v}\|_2, \qquad s_i^{\mathrm{GradNorm}} = \sum_b \frac{t_{i,b}}{N^{-1} \sum_{j=1}^N t_{j,b} + 10^{-12}}.$$

The normalization is recomputed at each scoring round and is also used for the single-block opacity-logit control. The spatial SNIP-GS and SynFlow-GS controls use

$$s_i^{\mathrm{product}} = \sum_v \sum_{b \in \{\boldsymbol{\mu}, \boldsymbol{\rho}\}} \sum_d \left| b_{i,d} \frac{\partial J_v}{\partial b_{i,d}} \right|. \tag{24}$$

The scale block is the stored log-scale $\boldsymbol{\rho}_i$; the renderer obtains activated scale as $\mathbf{s}_i = \exp(\boldsymbol{\rho}_i)$. The scoring product uses $\boldsymbol{\rho}_i$ and its gradient, with $J_v = \ell_{1,v}$ for SNIP-GS and $J_v = \mathbf{1}^\top R_v$ for SynFlow-GS. The GradNorm-GS (opacity logit) control differentiates the stored logit, which is distinct from a multiplicative gate on activated opacity.

GraSP-GS (Wang et al., 2020). For $g_v = \nabla_{\mathbf{m}} \ell_{1,v}|_{\mathbf{1}}$, the reported adaptation uses

$$s_i^{\mathrm{GraSP}} = -\sum_v \left[\frac{\nabla \ell_{1,v}(\mathbf{1} + \epsilon g_v) - \nabla \ell_{1,v}(\mathbf{1})}{\epsilon} \right]_i. \tag{25}$$

The finite-difference step is $\epsilon = 10^{-3}$, evaluated at unit base gates. The perturbed gates $\mathbf{1} + \epsilon g_v$ are evaluated without projection onto $[0, 1]$ and may exceed one. The estimate is a directional derivative around the base gate, not along the deletion path from 1 to 0.

Zen-GS (Lin et al., 2021). A small camera perturbation defines $o_v = \mathrm{mean}\,|R(\phi_v + \epsilon_v) - R(\phi_v)|$ and

$$s_i^{\mathrm{Zen}} = \sum_v \left| \partial o_v / \partial m_i |_{\mathbf{1}} \right|. \tag{26}$$

This adaptation attributes camera-induced image changes to individual Gaussian gates. One perturbation is sampled per view, with rotation scale 0.5 degrees and translation scale 0.02 in scene units, using a NumPy random generator with seed zero. Each score evaluation renders the original and perturbed cameras.

EPE-GS (Lopes et al., 2021). The reported construction uses camera-response signatures to weight the viewwise gate attribution. Camera centers are grouped into four clusters using Euclidean k-means with 25 iterations and random initialization seed zero. For each view, the rendered RGB difference under the same rotation and translation scales as Zen-GS is adaptively average-pooled to 32×32 and flattened to $\mathbf{u}_v$. Its signature is $\mathbf{w}_v = \mathbf{u}_v / (\|\mathbf{u}_v\|_2 + 10^{-12})$. For cluster c, stack its signatures as rows of W_c and subtract the column means to obtain X_c. Define

$$Q_c = X_c X_c^\top, \qquad (\Sigma_c)_{ab} = \frac{(Q_c)_{ab}}{\max\left(\sqrt{\max((Q_c)_{aa}, \epsilon_E) \max((Q_c)_{bb}, \epsilon_E)}, \epsilon_E \right)}, \quad \epsilon_E = 10^{-5}.$$

For clusters with at least two views, let $e_c = |\sum_{a,b} \log(|(\Sigma_c)_{ab}| + \epsilon_E)|$; otherwise set $e_c = 0$. Cluster weights are $\omega_c = e_c / (\mathrm{mean}_{c'}\, e_{c'} + 10^{-12})$. If $\sum_c e_c < 10^{-12}$, use uniform weights. The score is

$$s_i^{\mathrm{EPE}} = \sum_v \omega_{c(v)} \left| \frac{\partial o_v}{\partial m_i} \right|_{\mathbf{m}=\mathbf{1}},$$

where o_v is the full-resolution mean image difference defined for Zen-GS and $c(v)$ is the view's cluster. Signatures and gate attribution reuse the same perturbation for each view. A NumPy random state seeded with zero supplies per-view seeds; rotations use a random axis and an angle uniform in $[-0.5, 0.5]$ degrees, and translations have magnitude 0.02 along a random direction in the stored camera extrinsics. Cluster weights are held fixed during gate differentiation.

ZiCo-GS (Li et al., 2023). The spatial variant sums $|\operatorname{mean}_v g_{v,i,d}|/(\operatorname{std}_v g_{v,i,d} + \varepsilon)$ over position/log-scale coordinates. The gate variant uses the scalar opacity gate gradient. Invisible primitives have zero gradients and zero score.

Usage-GS and LightGaussian (Fan et al., 2024). For each pixel hit accepted by the counting rasterizer after its contribution and transmittance thresholds, the importance accumulator adds the Gaussian's activated opacity α_i. Let $H_{v,i}$ be the number of accepted pixel hits in view v. Usage-GS uses the opacity-weighted hit statistic $\mathrm{imp}_i = \alpha_i \sum_v H_{v,i}$, which measures rasterization usage rather than the alpha-blended color contribution. LightGaussian multiplies it by a volume factor: $s_i^{\mathrm{LG}} = (v_i/\kappa)^{0.1}\mathrm{imp}_i$, where $v_i = \prod_d s_{i,d}$ is activated volume. In the implementation, κ is the volume at zero-based index $\lfloor 0.9N \rfloor$ in descending order. This common positive normalization rescales scores without changing their ranking.

Fisher-GS. Fisher-GS (Abdelfattah et al., 2021; Theis et al., 2018) sums squared photometric gate derivatives, $s_i^{\mathrm{Fisher}} = \sum_v d_{v,i}^2$.

Jacov-GS. Let $d_{v,i} = \partial \ell_{1,v}/\partial m_i|_{\mathbf{1}}$ and $\mathbf{d}_v = (d_{v,1}, \ldots, d_{v,N})$. Normalization is over all Gaussians within each view; variability is then computed across views:

$$\widehat{d}_{v,i} = \frac{d_{v,i}}{\|\mathbf{d}_v\|_2 + 10^{-12}}, \qquad \bar{d}_i = \frac{1}{V}\sum_v \widehat{d}_{v,i}, \qquad s_i^{\mathrm{Jacov}} = \sqrt{\max\left(\frac{1}{V}\sum_v \widehat{d}_{v,i}^2 - \bar{d}_i^2,\, 0\right)}.$$

This population standard deviation turns cross-view gradient variation into a primitive-level ranking, replacing the source proxy's architecture-level spectral statistic.

NASWOT-GS (Mellor et al., 2021). For projected pixel radius $r_{i,v}^{\mathrm{pix}}$, define activity $a_{i,v} = \alpha_i \max(r_{i,v}^{\mathrm{pix}}, 0)^2$. In each view, τ_v is the median of strictly positive activities (the lower middle value for an even count). Binary codes are $M_{iv} = \mathbf{1}[a_{i,v} \geq \tau_v]$; a view with no positive activity has an all-zero column. With row i represented as column vector $\mathbf{b}_i \in \{0,1\}^V$, form

$$A = M^\top M + 10^{-3} I_V, \qquad \xi_i = \mathrm{clip}(\mathbf{b}_i^\top A^{-1} \mathbf{b}_i, 0, 1 - 10^{-6}), \qquad s_i^{\mathrm{NASWOT}} = -\log(1 - \xi_i).$$

This is a regularized binary-code removal marginal, with the numerical clamp applied before the logarithm; the fixed-feature identity is equation 18 in Appendix C.

First-order controls of the corrected scores. The signed L_1 first-order control extracts the negative-gradient component from the same scoring routine as Corrected L_1. The L_2 pair shares a squared-loss gate pass with optional squared image-Jacobian accumulation. These controls preserve the objective, normalization, and gate while omitting the correction from the ranking score (Table 4).

F Supplementary quantitative results

F.1 Extended proxy suite

Table 7 reports full metrics for the controls cited in Section 5. The spatial SNIP-GS and SynFlow-GS rows and PUP-α complete the coordinate comparison of Table 5. GradNorm-GS (opacity logit) shows that a coordinate involving opacity but not representing deletion loses 1.1 dB on Mip-NeRF 360 relative to the rank-equivalent gate score SNIP-GS. PUP (1×1) scores at full resolution and still remains below both corrected scores and all seven leading adapted proxies on every dataset, so PUP's position in Table 3 is not a consequence of reduced scoring resolution. ZiCo-GS (gate) stays below Random with the deletion coordinate, confirming that a consistency statistic does not become a deletion estimate by changing coordinates.

F.2 Per-scene test results

Table 8 reports per-scene PSNR under the main protocol. Both corrected scores exceed PUP on all thirteen scenes, with mean paired gains of 0.58 dB for Corrected L_1 and 0.54 dB for Corrected L_2, computed from unrounded results. Corrected L_1 also exceeds the two strongest gate-gradient proxies, SNIP-GS and SynFlow-GS, on all thirteen scenes.

Table 7: Reconstruction quality of coordinate, resolution, and proxy controls under the main pruning protocol.

	Mip-NeRF 360			Tanks & Temples			Deep Blending		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Coordinate and resolution controls									
SNIP-GS (spatial)	26.73	0.7850	0.2805	22.99	0.7998	0.2566	29.43	0.9029	0.2602
SynFlow-GS (spatial)	26.78	0.7793	0.2905	23.11	0.7997	0.2592	29.50	0.9019	0.2650
GradNorm-GS (opacity logit)	25.89	0.7631	0.2949	22.09	0.7809	0.2704	28.86	0.8899	0.2798
PUP (1×1)	26.15	0.7846	0.2744	22.64	0.8046	0.2460	29.37	0.9036	0.2557
PUP-α	26.83	0.7841	0.2695	23.00	0.8119	0.2356	29.48	0.9026	0.2550
Proxy control									
ZiCo-GS (gate)	25.44	0.7538	0.3164	22.10	0.7770	0.2840	28.77	0.8933	0.2820

Table 8: Per-scene test PSNR at 90% total pruning on Mip-NeRF 360, Tanks & Temples, and Deep Blending. Pruned models follow the main pruning protocol; the unpruned model is the original 30k checkpoint without additional refinement.

	Mip-NeRF 360									Tanks & Temples		Deep Blending	
Method	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	truck	train	drjohnson	playroom
3DGS (unpruned)	25.19	21.51	27.31	26.58	22.31	31.38	28.93	30.84	32.05	24.99	21.79	28.97	30.07
Random	24.28	20.03	25.88	25.27	22.14	29.77	27.03	28.48	29.13	24.05	20.79	28.47	29.50
LightGaussian	24.29	20.88	25.85	25.87	22.29	30.58	27.37	29.08	29.72	24.52	21.27	28.65	29.58
PUP	24.54	20.99	26.33	26.16	22.24	30.90	27.98	29.60	30.39	24.03	20.88	28.88	29.92
RAP	24.75	21.14	26.10	26.18	22.41	30.33	27.33	29.15	29.80	24.45	21.31	28.59	29.46
PUP-α	24.63	21.00	26.32	26.06	22.31	31.22	28.54	30.33	31.07	24.39	21.61	28.98	29.97
SNIP-GS	25.10	21.31	26.86	26.44	22.47	31.29	28.34	30.39	30.90	24.92	21.76	29.00	29.99
SynFlow-GS	25.13	21.34	26.87	26.47	22.46	31.30	28.41	30.44	30.97	24.98	21.77	29.06	30.01
GradNorm-GS	25.21	21.41	26.69	26.57	22.58	31.21	28.22	30.23	30.73	24.67	21.59	29.08	29.99
Zen-GS	25.19	21.42	26.93	26.56	22.51	31.32	28.37	30.05	30.82	24.94	21.74	29.07	29.97
Corrected L_1	25.19	21.44	27.06	26.60	22.57	31.37	28.59	30.52	31.06	25.02	21.82	29.07	30.04
Corrected L_2	25.06	21.41	27.00	26.54	22.38	31.36	28.57	30.55	31.09	25.00	21.73	29.11	30.05

F.3 Qualitative comparison

Figure 1 compares Corrected L_1 with PUP and LightGaussian under the main protocol. Corrected L_1 preserves background structure in kitchen and book boundaries in drjohnson that PUP loses.

F.4 Supplementary operating point

Table 9 extends the evaluation to 95% total pruning; all settings other than the first-round pruning rate follow the main protocol. Corrected L_1 achieves the highest PSNR and SSIM and the lowest LPIPS among the evaluated methods on all three datasets. Its PSNR margin over PUP grows from 0.59, 0.97, and 0.15 dB at 90% pruning to 0.89, 1.50, and 0.32 dB at 95%, and its margin over SynFlow-GS grows from 0.12, 0.04, and 0.02 dB to 0.22, 0.13, and 0.09 dB: the more aggressive the pruning, the more the exact deletion cost matters.

F.5 Repeatability

We conducted two independent runs of the same pruning and refinement protocol to assess the repeatability of Corrected L_1 and Corrected L_2. The mean absolute between-run differences in per-scene test PSNR were 0.0036 dB and 0.0052 dB, respectively, with maximum differences of 0.0151 dB and 0.0142 dB, more than an order of magnitude below the 0.58 and 0.54 dB mean gains over PUP.

F.6 Scoring cost

Table 10 reports the initial scoring time (Round 1) and the rescoring time after the first pruning and refinement stage (Round 2). Timing includes rendering required for scoring but excludes refine-

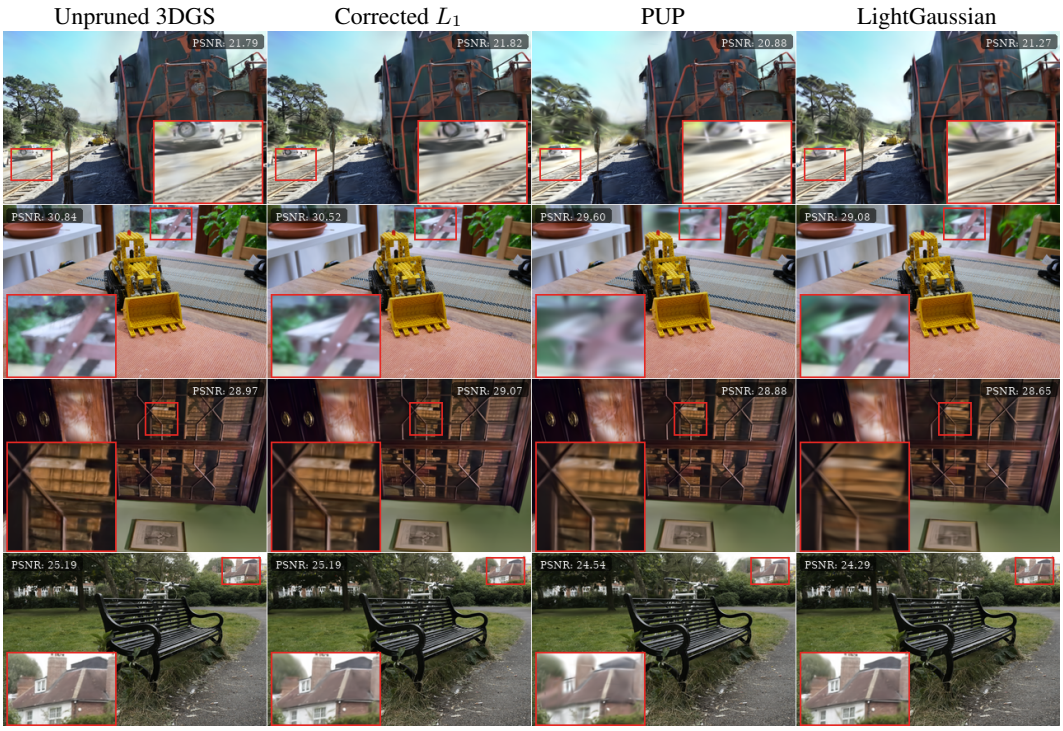


Figure 1: Visual comparison of unpruned and pruned reconstructions. Columns: Unpruned 3DGS, Corrected L_1, PUP, LightGaussian. Rows: train, kitchen, drjohnson, bicycle. The unpruned reference is rendered from the original 30k checkpoint without additional refinement; pruned models follow the main pruning protocol. Matched crops show local detail; PSNR labels report scene averages.

Table 9: Reconstruction quality at 95% total pruning. The two rounds remove 90% of the initial Gaussians and then 50% of the survivors, with 5k refinement steps after each round and rescoring before the second round. Metrics are averaged over scenes within each dataset after the second refinement stage.

	Mip-NeRF 360			Tanks & Temples			Deep Blending		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
LightGaussian	25.14	0.7157	0.3575	22.13	0.7534	0.3180	28.55	0.8795	0.3069
PUP	25.61	0.7530	0.3169	21.52	0.7619	0.2954	29.10	0.8965	0.2752
SNIP-GS	26.24	0.7612	0.3155	22.82	0.7865	0.2792	29.29	0.8958	0.2788
SynFlow-GS	26.28	0.7633	0.3134	22.89	0.7893	0.2779	29.33	0.8970	0.2766
Corrected L_1	26.50	0.7742	0.3007	23.02	0.7961	0.2708	29.42	0.9002	0.2701

ment, standalone evaluation, and subsequent PLY serialization. LightGaussian and RAP were timed separately under the same L40S protocol. For RAP, timing includes IVF feature construction with 128 neighbors and pretrained predictor inference, with device synchronization at both boundaries; predictor and PLY loading are excluded. Rescoring uses each method's checkpoint after 80% pruning and 5k refinement. Totals are computed from unrounded measurements and may differ from the sum of the displayed round times by 0.01 s.

Table 10: Per-round and total scoring times (seconds) for the main methods on one NVIDIA L40S, averaged over thirteen scenes.

Round	LightGaussian	PUP	RAP	SNIP-GS	SynFlow-GS	Corrected L_1	Corrected L_2
Round 1	2.24	19.64	8.26	5.00	4.89	5.53	5.49
Round 2	0.57	5.59	1.41	2.80	2.82	3.05	2.98
Total	2.81	25.22	9.67	7.79	7.71	8.58	8.47

G REPRODUCIBILITY DETAILS

Image resolution and evaluation split. Mip-NeRF 360 images are downsampled by factors of four for outdoor scenes and two for indoor scenes. Tanks & Temples and Deep Blending use the original image resolution. Every eighth view is held out for evaluation; rendering-based scores use only training views.