DELTAMERGER: LEARNING VISUAL MERGE COSTS FOR ZERO-SHOT GAUSSIAN SPLAT SIMPLIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce DeltaMerger, which simplifies pretrained 3D Gaussian Splatting representations by recursive pairwise merging with a learned merge cost, followed by a shared terminal refiner. Analytic merge costs measure the attribute discrepancy between two Gaussians and their proposed parent, but the visual effect of a merge also depends on visibility, compositing, and the distortions left by earlier merges. We supervise a compact shared predictor with rendered interventions at intermediate simplification states, measuring the immediate image change caused by replacing a candidate pair with its analytic parent and the signed change in error relative to the original rendering. The requested budget selects between two supervision regimes: the immediate change at 5–25% retention, and the change in fidelity to the original rendering, which can favor merges that partially repair earlier distortion, at 0.1–1%. Trained once on 64 DL3DV scenes, the frozen predictor reads only descriptors of the pair, its neighborhood, and its proposed parent, together with simplification progress and the requested budget, and transfers zero-shot to held-out DL3DV scenes and to four datasets never used for training, without images, cameras, rendering, or per-scene optimization. On 46 held-out scenes, DeltaMerger without refinement improves mean PSNR over NanoGS with the same candidate neighborhood at every retention budget, by 0.35 dB on average. Its shared refiner, trained once through differentiable rendering, corrects appearance and scale at fixed primitive count; the complete DeltaMerger improves on unrefined analytic selection by 1.12 dB, and learned selection keeps a 0.14 dB advantage when analytic selection receives its own refiner.

1 INTRODUCTION

3D Gaussian Splatting (3DGS) supports realistic scene capture and real-time novel-view synthesis (Kerbl et al., 2023), but its explicit primitives are costly to store and transmit, and pretrained representations may need to meet new memory budgets after reconstruction. An exported Gaussian model may not be accompanied by its reconstruction images and camera poses. This motivates simplification from the Gaussian PLY and target primitive count alone, with any learned model trained beforehand across scenes and frozen at deployment. RAP establishes this setting for learned pruning: a shared pretrained predictor scores each Gaussian from its attributes and neighborhood statistics and retains the highest-ranked primitives (Yang et al., 2026). Pruning discards the attributes of removed primitives, whereas pairwise merging replaces two primitives with one parent that approximates both, for the same count reduction. This motivates learning the visual cost of a pairwise replacement rather than the importance of an individual primitive.

Recursive pairwise merging repeatedly replaces a pair of Gaussians with a single representative. NanoGS constructs each parent analytically and selects neighboring pairs using a cost that combines geometric divergence with appearance discrepancy (Xiong et al., 2026). Attribute similarity does not directly measure a merge's visual effect, which also depends on projection, visibility, and compositing with surrounding primitives. Recursive merging further changes this context: earlier parents become later candidates, and preserving the current rendering may preserve accumulated distortion.

We introduce DeltaMerger, which learns pair-selection costs while keeping candidate generation and analytic parent construction fixed. A shared predictor encodes each pair, its neighborhood, and

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.

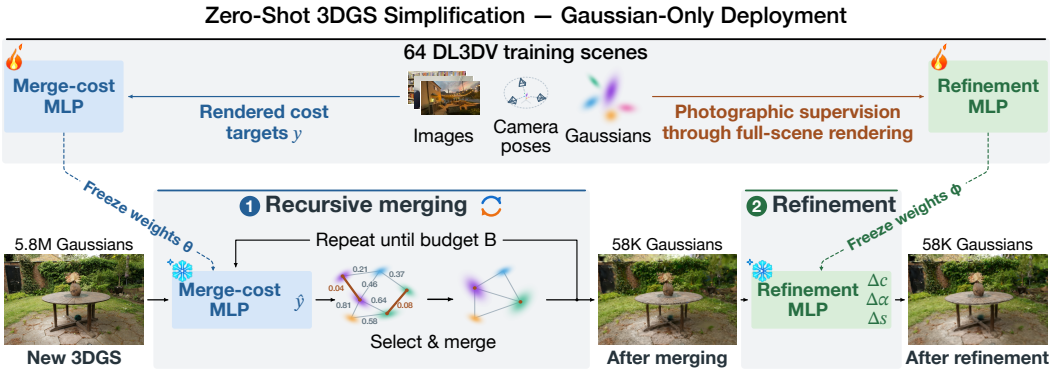

Figure 1: DeltaMerger overview. **Top (training, once):** on 64 DL3DV scenes, the merge-cost MLP is supervised by rendered-intervention cost targets y, and the shared refinement MLP by photographs through full-scene rendering. **Bottom (deployment, zero-shot):** both networks are frozen and receive only the Gaussians of a new scene and the requested budget. (1) The merge-cost MLP scores candidate pairs ($\hat{y}$); low-cost disjoint pairs are replaced by their analytic parents, and passes repeat until the budget B is reached. (2) The shared refiner corrects the color, opacity, and scale $(\Delta c, \Delta\alpha, \Delta s)$ of the final representatives at fixed primitive count, using statistics of their source groups retained by the pipeline. Example: Mip-NeRF 360 Garden, 5.8M to 58K Gaussians (1% retention); edge costs are illustrative.

proposed parent through structured Gaussian descriptors, conditioned on simplification progress and budget. It is trained on full-scene rendered interventions at intermediate states: each candidate pair is independently replaced by its analytic parent while all other primitives remain fixed. Learning helps in two ways. An analytic cost compares children and parent in attribute space, so it does not see how much of a merge's effect survives projection and compositing; a learned estimate of the immediate image change does. Child–parent discrepancies also do not model how a merge interacts with the distortion left by earlier merges, although under strong compression a merge can partially repair it; this interaction depends on the accumulated rendering error, which is unavailable at deployment, so we learn a transferable proxy for it from descriptors (Section 3.2).

Trained once on 64 DL3DV scenes, the frozen predictor transfers zero-shot to 46 held-out scenes from five datasets, without source images, camera poses, rendering, or per-scene adaptation. A predictor trained on the immediate change alone already improves on analytic selection by 0.45 dB at 5–25% retention; the fidelity signal matters at 0.1% and 1%. A shared refiner, trained once against training photographs through the renderer, then corrects color, opacity, and scale at fixed primitive count. Our contributions are:

- A transferable visual merge-cost predictor for recursive pair selection with analytic parent construction.
- Rendered-intervention supervision with a budget-dependent target (immediate image change at 5–25% retention, signed original-rendering fidelity at 0.1–1%), whose decomposition separates the immediate change from its interaction with accumulated error, which child–parent discrepancies do not model.
- A shared terminal refiner applied in one asset-only pass (from the Gaussians alone) at fixed primitive count, evaluated separately from the learned merge cost.
- A zero-shot evaluation on 46 scenes from five datasets, with matched ablations of the supervision target, objective, and descriptors, and a matched comparison of learned and analytic selection under refinement.

2 Related Work

Gaussian splatting simplification. Gaussian simplification reduces primitive count, complementing attribute quantization and coding. Some methods encourage compactness during reconstruction through learned masks (Lee et al., 2024; Liu et al., 2025), pruning or sparsification during train-

ing (Papantonakis et al., 2024; Girish et al., 2024; Zhang et al., 2025), budget-aware densification (Mallick et al., 2024), or densification followed by Gaussian subset selection (Fang & Wang, 2024). Others reduce an existing representation: LightGaussian (Fan et al., 2024) and Trimming the Fat (Ali et al., 2024) combine importance-guided pruning with quality recovery. Level-of-detail methods build per-scene hierarchies of merged parent Gaussians (Kerbl et al., 2024) or multi-resolution anchors (Ren et al., 2025). These methods rely on scene-specific rendering or reconstruction signals. Merging instead combines primitives into new representatives, as greedy edge collapse ordered by quadric error metrics does for meshes (Garland & Heckbert, 1997). GHAP performs geometric reduction through an optimal-transport objective, followed by optimization of color and opacity (Wang et al., 2025); we compare with its geometric stage alone (Appendix B.11). NanoGS recursively merges neighboring pairs using analytic moment matching, directly on pretrained Gaussian representations without training images or per-scene optimization (Xiong et al., 2026). CVT-GS constructs spatial support cells through centroidal Voronoi tessellation and uses a shared network, trained under local rendering supervision, to predict a representative Gaussian for each cell (Li et al., 2026). We instead learn pair-selection costs and retain analytic parents; our shared refiner corrects final attributes with a shared network trained once through full-scene rendering, without fitting to the deployment scene.

Importance scores and merge costs. Classical Gaussian mixture reduction includes pairwise moment matching (Salmond et al., 1988) with selection criteria based on integrated squared error (Williams & Maybeck, 2006) or bounds on Kullback–Leibler divergence (Runnalls, 2007); NanoGS combines a geometric discrepancy between a pair's mixture and its replacement with an appearance discrepancy (Xiong et al., 2026). Attribute agreement does not directly capture visibility or compositing effects. For pruning, PUP-3DGS and Speedy-Splat estimate importance from reconstruction-error sensitivity (Hanson et al., 2025b;a); REFINE avoids rasterization but still uses camera parameters (Chen et al., 2026), whereas camera-agnostic pruning derives confidence from Gaussian descriptors (Fasogbon et al., 2026). RAP learns per-Gaussian importance from intrinsic attributes and neighborhood statistics under rendering supervision, then applies a shared frozen model directly to a PLY, without source images, camera poses, rendering, or per-scene fitting (Yang et al., 2026). We share this Gaussian-only deployment setting but predict a different quantity: RAP scores primitives for subset selection, whereas DeltaMerger scores pair-to-parent replacements that combine attributes from both children, learned from full-scene interventions that measure immediate rendering change and signed original-rendering fidelity.

3 DeltaMerger

DeltaMerger learns a visual cost for recursive Gaussian merging from rendered interventions. The shared predictor transfers to new scenes using Gaussian descriptors, while candidate generation and analytic parent construction remain fixed during recursive simplification. A shared refiner subsequently adjusts terminal attributes at fixed primitive count (Section 3.3). We call the complete system DeltaMerger and, without the refiner, DeltaMerger w/o refinement.

3.1 Preliminaries: Recursive Gaussian Merging

Gaussian representation. A 3D Gaussian Splatting representation $\mathcal{G}_0$ contains N_0 primitives $g_i = (\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \alpha_i, \mathbf{h}_i)$, parameterized by their center, covariance, opacity, and spherical-harmonic appearance coefficients (Kerbl et al., 2023). For retention ρ, we seek $B = \max(1, \lfloor \rho N_0 \rfloor)$ primitives while preserving visual fidelity.

Recursive pairwise merging. At state $\mathcal{G}_t$, edges of a k-nearest-neighbor graph over Gaussian centers define candidates $\mathcal{E}_t$. For a pair $e = \{i, j\}$, the operator $\mathcal{M}$ replaces its children with one parent:

$$\mathcal{G}_t^e = \big(\mathcal{G}_t \setminus \{g_i, g_j\}\big) \cup \{\mathcal{M}(g_i, g_j)\}. \tag{1}$$

Each accepted replacement reduces the count by one, as deleting one primitive does. We use the analytic moment-matching construction of NanoGS (Xiong et al., 2026), with the complete parent equations provided in Appendix B.3. Algorithm 1 (Appendix B.1) greedily merges disjoint pairs with low cost $c(\mathcal{G}_t, e; B)$, rebuilding the graph after each pass until exactly B primitives remain.

Local candidates reduce each pass from $O(N_t^2)$ to $O(kN_t)$ pairs, where $N_t = |\mathcal{G}_t|$; these local, irreversible decisions do not guarantee a globally optimal simplification.

3.2 Learning a Transferable Visual Merge Cost

3.2.1 Candidate Representation

We encode children, their neighborhood, and the proposed analytic parent with descriptors shared across scenes. Sums and absolute differences summarize child attributes, local spacing, appearance coherence, and opacity. Further features capture pair geometry, spherical-harmonic differences and degree-wise energy, pair mass, and parent-induced changes in mass, scale, and anisotropy (Appendix B.6).

The predictor also receives the current simplification state and requested budget:

$$\mathbf{u}_t = \big[r_t, \log r_t, \log q, \log \rho\big], \qquad r_t = \frac{N_t}{N_0}, \qquad q = \frac{N_p}{N_0}, \tag{2}$$

where N_p is the count after prefiltering; r_t, q, and ρ encode current, post-prefiltering, and target retention, respectively.

A shared normalizer fitted on training candidates maps these descriptors to $\mathbf{z}_{t,e}$, and a compact MLP predicts $\hat{y}_{t,e} = f_\theta(\mathbf{z}_{t,e})$. Its linear output accommodates the signed fidelity target, under which repairs are negative; lower predictions indicate preferred merges.

3.2.2 Rendered-Intervention Supervision

Training interventions. Across training scenes and budgets, we sample candidates at intermediate states and independently replace each pair with its analytic parent, holding all other primitives fixed. For camera v, we render the full scene before and after replacement, $\mathbf{I}_t^{(v)} = \mathcal{R}(\mathcal{G}_t, v)$ and $\mathbf{I}_{t,e}^{(v)} = \mathcal{R}(\mathcal{G}_t^e, v)$, preserving projection, visibility, and alpha compositing. The reference $\mathbf{I}_0^{(v)} = \mathcal{R}(\mathcal{G}_0, v)$ is rendered before prefiltering.

Immediate change and original-rendering fidelity. Over training views $\mathcal{V}$, with $P = \sum_{v\in\mathcal{V}} 3H_vW_v$ color samples, the original-rendering error is

$$\mathcal{E}^0(\mathcal{G}) = \frac{1}{P}\sum_{v\in\mathcal{V}} \left\| \mathcal{R}(\mathcal{G}, v) - \mathbf{I}_0^{(v)} \right\|_F^2. \tag{3}$$

We then define the two supervision signals:

$$\mathcal{D}_{t,e} = \frac{1}{P}\sum_{v\in\mathcal{V}} \left\| \mathbf{I}_{t,e}^{(v)} - \mathbf{I}_t^{(v)} \right\|_F^2, \tag{4}$$

$$\Delta\mathcal{E}_{t,e}^0 = \mathcal{E}^0(\mathcal{G}_t^e) - \mathcal{E}^0(\mathcal{G}_t). \tag{5}$$

Immediate change versus accumulated error. Let $\boldsymbol{\varepsilon}_t^{(v)} = \mathbf{I}_t^{(v)} - \mathbf{I}_0^{(v)}$ be the error accumulated by prefiltering and earlier merges, and $\boldsymbol{\delta}_{t,e}^{(v)} = \mathbf{I}_{t,e}^{(v)} - \mathbf{I}_t^{(v)}$ the perturbation caused by merge e. Expanding Equation 5 gives

$$\Delta\mathcal{E}_{t,e}^0 = \mathcal{D}_{t,e} + \mathcal{K}_{t,e}, \qquad \mathcal{K}_{t,e} = \frac{2}{P}\sum_{v\in\mathcal{V}} \big\langle \boldsymbol{\varepsilon}_t^{(v)}, \boldsymbol{\delta}_{t,e}^{(v)} \big\rangle_F. \tag{6}$$

The immediate change $\mathcal{D}_{t,e} \geq 0$ measures how far a merge moves the current rendering, whereas the signed cross term $\mathcal{K}_{t,e}$ is negative when the perturbation opposes the accumulated error. Hence $\Delta\mathcal{E}_{t,e}^0 < 0$ exactly when $\mathcal{K}_{t,e} < -\mathcal{D}_{t,e}$, that is, when the merge partially repairs earlier distortions; about 30% of our sampled training candidates do so at every budget, and the median of $|\mathcal{K}_{t,e}|/\mathcal{D}_{t,e}$ lies between 3.2 and 3.6. Analytic merge costs compare children with their parent and act as proxies for $\mathcal{D}_{t,e}$; they do not model the cross term, which depends on $\boldsymbol{\varepsilon}_t$. Since $\boldsymbol{\varepsilon}_t$ is unavailable at deployment, this motivates learning a transferable proxy for the visual effect of a merge from descriptors of the pair, its neighborhood, its proposed parent, and the simplification state (Appendix B.4).

Budget-dependent target. We combine the two signals into a single visual merge cost:

$$Y_{t,e} = \big(1 - \lambda(\rho)\big)\mathcal{D}_{t,e} + \lambda(\rho)\Delta\mathcal{E}^0_{t,e} = \mathcal{D}_{t,e} + \lambda(\rho)\,\mathcal{K}_{t,e}. \tag{7}$$

The retention-dependent weight $\lambda(\rho) \in [0, 1]$ selects what the cost learns. With $\lambda = 0$ it learns the immediate change, which, unlike attribute discrepancies, reflects visibility and compositing; with $\lambda = 1$ it learns the change in original-rendering fidelity, including repairs; $0 < \lambda < 1$ would add a proximal penalty on changes to the current state (Appendix B.4). Training uses five supervised budgets, where λ is 0 or 1; the network's score is conditioned directly on the requested budget. We use $\lambda = 1, 1, 0, 0, 0$ at $0.1, 1, 5, 10, 25\%$ retention, selected on a held-out validation set (Appendix B.7).

3.2.3 Training Objective

For each group g of candidates sharing a state and budget, we normalize targets as $y_{t,e} = Y_{t,e}/s$, with a shared positive training-data scale s, and combine Huber regression with pairwise ranking:

$$\mathcal{L} = \sum_g w_g \left(\mathcal{L}^{(g)}_{\text{reg}} + \lambda_{\text{rank}}\mathcal{L}^{(g)}_{\text{rank}}\right). \tag{8}$$

Regression fits target values; weighted logistic ranking orders distinct targets within each group. Weights w_g balance scenes, budgets, state-collection policies, and sampled groups (Appendix B.7).

3.3 Zero-Shot Recursive Simplification

With weights and normalization frozen, deployment requires only the Gaussian representation and target count, without source images, camera poses, rendering, or per-scene optimization; the accumulated error ε_t is never measured; the cost only learns its effects. After prefiltering, we instantiate Algorithm 1 with $c(\mathcal{G}_t, e; B) = \hat{y}_{t,e}$, recomputing descriptors and scores after each pass so that later decisions account for previously merged parents and their updated context. Training labels describe individual interventions; their predicted costs prioritize disjoint merges without assuming additive rendering effects.

Shared terminal refinement. After reaching the requested budget, a shared refiner corrects the final representatives in one pass, at fixed primitive count. For each representative containing more than one source primitive, it predicts bounded corrections to its SH-DC color, opacity, and three scales from a 40-dimensional asset-only descriptor of the representative, its source group, and its eight nearest final neighbors, conditioned on the budget through FiLM (Perez et al., 2018); count, group membership, centers, rotations, higher-order SH coefficients, and singletons remain fixed. Like the merge-cost predictor, it is a single network shared across scenes and budgets (7,751 parameters) and trained once: we simplify each training scene at every budget with the frozen planner (recursive merging with the learned cost), apply the refiner, render the full scene from training cameras, and minimize the RGB error against the training photographs through the renderer. At deployment, one pass produces a standard Gaussian asset without photographs, cameras, rendering, per-scene optimization, or per-frame network evaluation (Appendix B.8).

4 Experiments

We evaluate simplification quality, zero-shot transfer, and component contributions. Table 1 reports DeltaMerger with and without terminal refinement, as do Figures 2 and 3, and Table 2 compares learned and analytic selection, each followed by its own refiner.

4.1 Implementation Details and Evaluation Protocol

Implementation. We train on 49,152 interventions from 64 DL3DV scenes (Ling et al., 2024): 2,048 state–budget groups of 24 candidates, each evaluated over eight views, collected along both analytic and predictor-guided trajectories (Appendix B.5). The predictor is a 35-input MLP with hidden widths $64, 32$ (4,417 parameters), trained with 200 full-batch AdamW updates and $\lambda_{\text{rank}} = 4$ for three independent seeds; λ is defined at the five training budgets, and other budgets use the budget-conditioned score directly. Each pass recomputes four candidate and eight context neighbors,

and every recursive parent uses native NanoGS moment matching with degree-three SH. For each predictor seed, one shared refiner is trained on the frozen predictor's outputs (6,000 updates, about 15 minutes on one RTX 4090). Training, development, and evaluation splits are disjoint, although some evaluation scenes were inspected during development (Appendix B.9).

Evaluation. We evaluate 46 scenes from DL3DV, Deep Blending (Hedman et al., 2018), Mip-NeRF 360 (Barron et al., 2022), NeRF Synthetic (Mildenhall et al., 2020), and Tanks and Temples (Knapitsch et al., 2017). Baselines are NanoGS (Xiong et al., 2026) with $k = 4$ candidate neighbors, which uses our candidate rule and neighborhood size and serves as the primary reference, NanoGS with its official default $k = 16$, frozen RAP (Yang et al., 2026), and GHAP's geometric reduction without its photometric fine-tuning (GHAP, no FT) (Wang et al., 2025), which consumes only Gaussian attributes (Appendix B.11). All methods receive identical source representations at the five training retentions ($0.1\%, 1\%, 5\%, 10\%, 25\%$) and keep their own preprocessing and count rounding. We compute PSNR, SSIM, and VGG-LPIPS on held-out images with identical cameras, renderer, and image processing, average views per scene and then the three training seeds, and weight scenes equally (Appendix B.9).

4.2 Quantitative Results and Zero-Shot Transfer

Simplification quality. DeltaMerger w/o refinement improves PSNR and SSIM over NanoGS ($k = 4$) at every budget and LPIPS at every budget except 1% (Table 1), with a mean PSNR gain of 0.354 dB across budgets. At 10%, the gain reaches 0.524 dB, with improvements on 45 of 46 scenes. Against the official $k = 16$ configuration, DeltaMerger w/o refinement improves PSNR by 0.30 to 0.75 dB from 1% to 25% retention, with better SSIM and LPIPS, but is 0.03 dB lower at 0.1%, where $k = 16$ is better on 24 of 46 scenes. The wider candidate neighborhood thus helps NanoGS only under the most aggressive budget and lowers its quality at every other budget, whereas the learned cost improves selection within the fixed $k = 4$ neighborhood at every budget. GHAP (no FT) has lower mean PSNR and SSIM and higher mean LPIPS than both NanoGS and DeltaMerger w/o refinement at every requested retention (Table 1). These results characterize geometric reduction without photometric recovery and do not establish the quality of the complete GHAP pipeline.

Contribution of terminal refinement. On the 230 paired scene–budget cells, with each of the three predictor seeds refined by its own refiner, refinement adds 0.764 dB PSNR, 0.00984 SSIM, and reduces LPIPS by 0.01249 relative to the same unrefined predictor (Appendix Table 11). The largest average PSNR gain occurs at 5% retention. Together with the merge-only gain of learned over analytic selection (0.35 dB), this refinement gain makes up the 1.12 dB gain of the complete system over analytic selection (Table 2). The gain is smallest at 25% retention (0.467 dB), and PSNR regresses on 8 of the 230 cells (Appendix B.13).

Transfer across scenes and datasets. Figure 2 separates within-dataset transfer to held-out DL3DV scenes from transfer to four external datasets. Without refinement and averaging three seeds, budget-averaged PSNR gains over NanoGS ($k = 4$) range from 0.257 to 0.468 dB across datasets, despite regressions on some scenes.

Failure cases. Losses to NanoGS ($k = 4$) concentrate at the lowest budgets: without refinement, the learned cost is worse on 13 of 46 scenes at 0.1% and 9 at 1%, but on at most four from 5% upward (Appendix B.10).

Qualitative results. Figure 3 compares NanoGS ($k = 4$) and DeltaMerger with and without refinement on held-out views of evaluation scenes. Appendix C adds eight scenes at all five budgets with fixed views.

4.3 Ablation Studies

Table 3 compares variants of the selected predictor on the 46 evaluation scenes, over five budgets and three seeds, without refinement; only the ablated component changes.

Table 1: Quality on the same 46 scenes from five datasets at each requested retention. DeltaMerger w/o ref. isolates the learned merge cost; DeltaMerger adds the shared refiner, which uses no images at deployment. GHAP (Wang et al., 2025) (no FT) omits its photometric recovery (Appendix B.11). We average views, three DeltaMerger training seeds (each with its own refiner), then scenes equally; Mean averages the five budgets, and paired 95% intervals of the gains are in Figure 2 and Table 2. NanoGS (Xiong et al., 2026) ($k = 4$) uses our candidate rule and neighborhood size; $k = 16$ is its official default. Best results are **bold**, second-best underlined.

Original (100%): PSNR 28.454 dB, SSIM 0.8746, LPIPS 0.2514

Method	Metric	0.1%	1%	5%	10%	25%	**Mean**
NanoGS ($k = 4$)	PSNR ↑	16.858	19.791	21.941	23.130	25.208	21.385
	SSIM ↑	0.6235	0.6625	0.7122	0.7448	0.8034	0.7093
	LPIPS ↓	0.5975	0.5273	0.4513	0.4101	0.3409	0.4654
NanoGS ($k = 16$)	PSNR ↑	17.033	19.679	21.729	22.908	24.939	21.258
	SSIM ↑	0.6238	0.6612	0.7069	0.7383	0.7983	0.7057
	LPIPS ↓	0.5953	0.5298	0.4568	0.4163	0.3460	0.4689
RAP	PSNR ↑	10.490	14.971	19.146	21.409	25.003	18.204
	SSIM ↑	0.4106	0.5667	0.6676	0.7251	0.8131	0.6366
	LPIPS ↓	0.6365	0.5460	0.4509	0.4005	0.3219	0.4712
GHAP (no FT)	PSNR ↑	7.642	10.716	15.874	18.704	22.715	15.130
	SSIM ↑	0.1727	0.3745	0.5508	0.6305	0.7392	0.4935
	LPIPS ↓	0.6644	0.6191	0.5254	0.4709	0.3883	0.5336
DeltaMerger w/o ref.	PSNR ↑	17.006	19.983	22.380	23.654	25.673	21.739
	SSIM ↑	0.6249	0.6649	0.7182	0.7542	0.8143	0.7153
	LPIPS ↓	0.5966	0.5278	0.4457	0.4004	0.3271	0.4595
DeltaMerger	PSNR ↑	**17.618**	**20.891**	**23.351**	**24.517**	**26.140**	**22.503**
	SSIM ↑	**0.6281**	**0.6730**	**0.7314**	**0.7690**	**0.8242**	**0.7252**
	LPIPS ↓	**0.5909**	**0.5163**	**0.4280**	**0.3829**	**0.3171**	**0.4470**

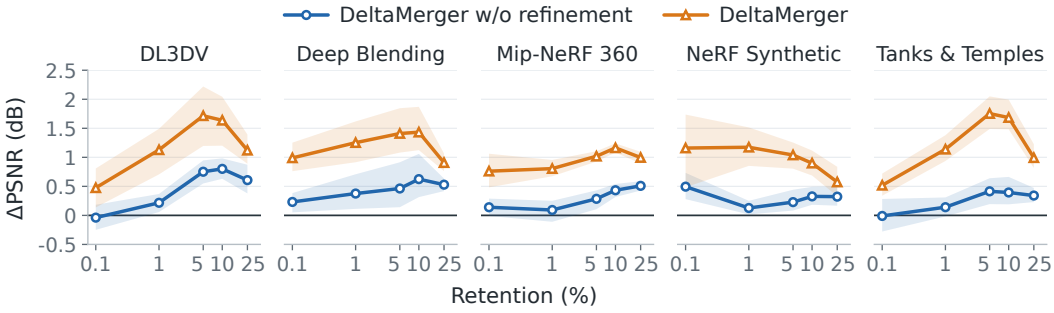

Figure 2: PSNR gain over NanoGS ($k = 4$) of DeltaMerger without refinement (blue) and with each seed's own refiner (orange), averaging three fixed predictor seeds, then scenes ($n = 10, 9, 9, 8, 10$). Shading connects pointwise 95% paired scene-bootstrap intervals (20,000 resamples), conditional on the fitted models; interpolation between budgets is visual only.

Learned versus analytic selection. Replacing predicted scores with the analytic criterion while fixing prefiltering, candidates, parents, and disjoint-pair selection isolates the scoring rule. It reduces PSNR at every budget, by 0.14 to 0.52 dB.

Selection with matched refinement. Terminal refinement could repair errors that the learned cost avoids. We therefore train a separate refiner on the outputs of each planner with the same recipe and evaluate the four combinations on the same 230 cells (Table 2). Learned selection improves on analytic selection by 0.35 dB without refinement ([0.29, 0.43]; 43 of 46 scenes) and by 0.14 dB when each planner is followed by its own refiners ([0.11, 0.19]; 40 of 46 scenes), mostly at 5%–

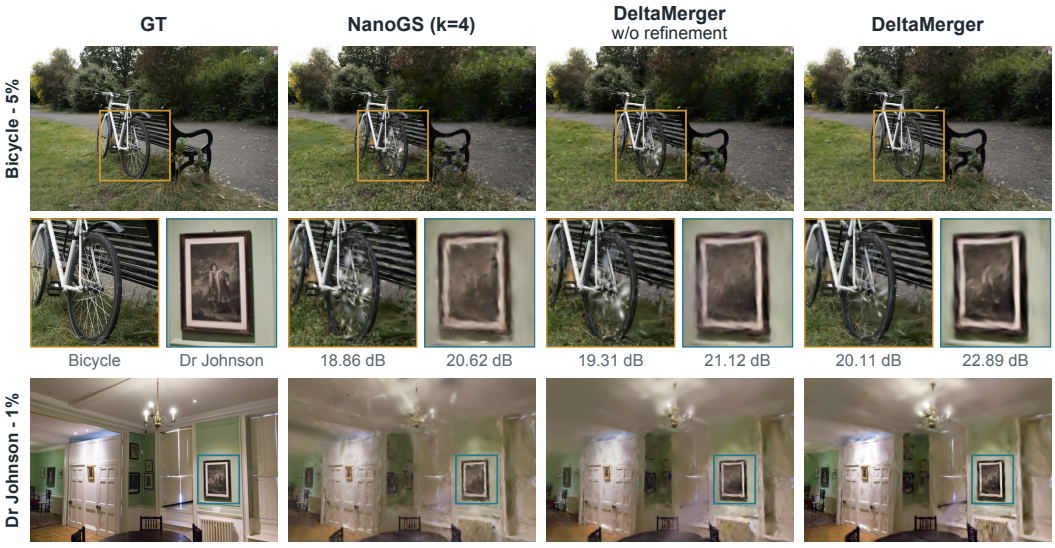

Figure 3: Bicycle (5%, top) and Dr Johnson (1%, bottom), using the seed-0 predictor and its refiner. Middle: their respective crops (left/right), with full-view PSNR. Views were selected near the 70th percentile of the unrefined gain over NanoGS, separately by dataset and budget.

Table 2: PSNR (dB) on 230 paired cells (46 scenes). A_0 is our pipeline with the NanoGS analytic cost; it matches NanoGS ($k = 4$) in Table 1 within 0.005 dB PSNR and 0.0003 SSIM and LPIPS at every budget. Learned: mean of three predictor/refiner pairs; analytic: mean of three refiner seeds. Each planner's refiners use its own outputs and the same recipe. Interaction is $(L_1 - L_0) - (A_1 - A_0)$. Contrasts report 95% paired scene-bootstrap intervals; n_+ counts positive budget-averaged scene differences.

Variant / contrast	0.1%	1%	5%	10%	25%	**Mean [95% CI]**	n_+
Analytic (A_0)	16.86	19.79	21.94	23.13	25.21	21.39	
+ refinement (A_1)	17.79	20.77	23.06	24.21	25.96	22.36	
Learned (L_0)	17.01	19.98	22.38	23.65	25.67	21.74	
+ refinement (L_1)	17.62	20.89	23.35	24.52	26.14	22.50	
Selection ($L_0 - A_0$)	+0.14	+0.19	+0.44	+0.52	+0.47	+0.35 [+0.29, +0.43]	43/46
Refined selection ($L_1 - A_1$)	−0.17	+0.12	+0.29	+0.30	+0.18	+0.14 [+0.11, +0.19]	40/46
Analytic refiner ($A_1 - A_0$)	+0.93	+0.98	+1.12	+1.08	+0.75	+0.97 [+0.86, +1.09]	44/46
Learned refiner ($L_1 - L_0$)	+0.61	+0.91	+0.97	+0.86	+0.47	+0.76 [+0.65, +0.87]	45/46
Interaction	−0.32	−0.07	−0.15	−0.22	−0.29	−0.21 [−0.26, −0.16]	5/46
Full system ($L_1 - A_0$)	+0.76	+1.10	+1.41	+1.39	+0.93	+1.12 [+1.01, +1.24]	45/46

25% retention; at 0.1% the refined analytic planner is better (−0.17 dB). The two stages overlap: refinement gains more on analytic outputs (0.97 dB) than on learned ones (0.76 dB).

Supervision target. We compare immediate change $\mathcal{D}_{t,e}$, original-rendering fidelity $\Delta\mathcal{E}^0_{t,e}$, and their combination, keeping the reference comparison weights β (Appendix B.7) and changing only the target. Immediate change alone ($\lambda = 0$) loses 0.25 and 0.16 dB at 0.1% and 1% but stays within 0.09 dB from 5% upward; fidelity alone ($\lambda = 1$) is within 0.10 dB of the selected predictor at 0.1% and 1% but loses 0.15–0.30 dB from 5% upward. In the studied protocol, this supports the fidelity signal at 0.1–1% and the immediate change at 5–25% retention. Learning from rendered changes alone already gives much of the gain: the immediate-change model improves on analytic selection by 0.45 dB at 5–25% (paired 95% interval [0.36, 0.54]; 45 of 46 scenes), but not at 0.1% and 1% (−0.11 and +0.03 dB).

Training objective. With targets and descriptors fixed, Huber-only regression matches the combination at 0.1% but loses 0.06–0.35 dB at the other budgets, while ranking alone is within 0.04 dB

Table 3: ΔPSNR (dB) from the selected predictor ($\lambda = (1, 1, 0, 0, 0)$) on the 46 evaluation scenes, averaged over three seeds, without refinement; positive values favor the variant. Target ablations use constant λ. Mean over budgets with 95% scene-bootstrap intervals (46 scenes).

Component	Variant	0.1%	1%	5%	10%	25%	Mean [95% CI]
Selection	Analytic	−0.144	−0.188	−0.440	−0.524	−0.466	−0.352 [−0.427, −0.284]
Target	Immediate $\mathcal{D}$ ($\lambda = 0$)	−0.254	−0.155	−0.090	−0.019	+0.023	−0.099 [−0.137, −0.062]
	Fidelity $\Delta\mathcal{E}^0$ ($\lambda = 1$)	−0.053	−0.094	−0.153	−0.248	−0.300	−0.170 [−0.202, −0.136]
Objective	Huber only	−0.004	−0.058	−0.131	−0.237	−0.350	−0.156 [−0.185, −0.128]
	Ranking only	−0.038	−0.034	−0.027	+0.009	−0.002	−0.018 [−0.039, +0.003]

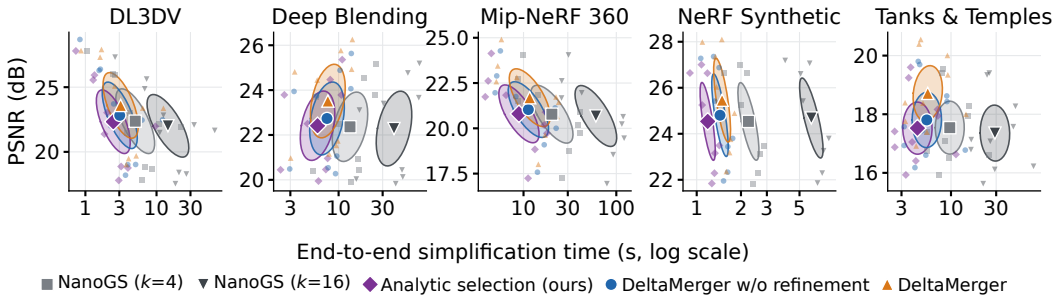

Figure 4: Speed and quality of the timed outputs per evaluation scene (small markers; our methods with seed-0 models): time is the geometric mean over the five budgets, PSNR the arithmetic mean. Large markers: centroids over scenes, with 95% scene-bootstrap confidence ellipses.

(mean −0.02 dB, interval including zero), so this ablation shows no clear benefit of the regression term on the evaluated panel.

Input representation. Removing the mass and proposed-parent descriptors costs up to 0.34 dB at 5–25% retention; the other input ablations change PSNR by at most 0.08 dB (Table 4).

4.4 Computational Cost

Figure 4 times complete PLY-to-PLY processes on four pinned cores, one at a time (Appendix B.12). With an optimized CPU build, DeltaMerger takes 7.0 s on average, against 12.0 s for the official optimized NanoGS ($k = 4$; $k = 16$: 35.8 s), at the same mean peak memory (1.5 GB); learned selection takes 1.24–1.25× the time of analytic selection, and the refiner adds 0.2 s. Offline (Table 10), label collection is estimated at 48 minutes on 24 cores and one RTX 5090 (CPU-bound), refiner training states about 5 CPU-job-hours, and network training under 20 minutes on one RTX 4090.

5 Conclusion

DeltaMerger learns transferable visual merge costs from rendered interventions while retaining analytic parent construction during recursive merging, with two supervision regimes: the immediate image change at 5–25% retention, and fidelity to the original rendering, including repairs, at 0.1–1%. The frozen predictor improves mean PSNR over NanoGS without rendering or per-scene optimization at deployment, and its shared refiner adds an asset-only correction at fixed primitive count.

Limitations. With matched refiners, learned selection keeps a 0.14 dB advantage but loses at 0.1% retention. The refiner needs source-group statistics, which a standalone simplified PLY may not contain. Both networks are trained and evaluated at five retentions only, and some evaluation scenes were seen in development. Merge decisions are greedy and irreversible.

Reproducibility Statement

Section 4.1 and Appendices B.2–B.7 specify the descriptors, rendered-intervention supervision, sampling, architecture, objective, and optimization of the merge-cost predictor; Appendix B.3 gives the analytic parent equations and Appendix B.8 the refiner. Appendix B.9 gives the evaluation protocol, the source representations, per-scene results (Table 6), and the training scene identifiers (Table 7); Appendix B.11 documents the baseline adaptations and Appendix B.12 the timing protocol and hardware. Unless stated otherwise, quantitative quality results average three independently trained seeds, and refined results pair each planner with its own refiner; qualitative examples and timings use seed 0. Confidence intervals resample scenes within the reported evaluation subset and are conditional on the fitted models. Code, trained weights, training labels, and per-cell results will be released upon publication.

AI Use Statement

We used OpenAI Codex and Anthropic Claude to assist with manuscript drafting and revision. Codex also assisted with literature search and reference checking, code implementation and debugging, and experiment orchestration. The core methodological ideas and equations were developed by the authors. The authors reviewed the manuscript and checked the references and code. We take responsibility for the final content, including all AI-assisted text, code, and analyses.

Ethics Statement

DeltaMerger is a post-hoc simplification method for Gaussian Splatting representations. It does not introduce new data collection, generate novel scene content, or rely on personally identifiable information beyond what may already be present in the input scenes. As with other 3D reconstruction and rendering methods, users should ensure that the source images and reconstructed scenes are collected, processed, and shared with appropriate consent and in compliance with applicable privacy and copyright requirements.

References

Muhammad Salman Ali, Maryam Qamar, Sung-Ho Bae, and Enzo Tartaglione. Trimming the fat: Efficient compression of 3D Gaussian splats through pruning. In *BMVC*, 2024.

Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022.

Zhang Chen, Shuai Wan, Mengting Yu, Fuzheng Yang, and Junhui Hou. REFINE: Super-efficient 3D Gaussian Splatting pruning via rendering-free primitive importance. In *ECCV*, 2026.

Hanjun Dai, Elias Khalil, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Oren Dovrat, Itai Lang, and Shai Avidan. Learning to sample. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2760–2769, 2019.

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 *NeurIPS*, 2024.

Guangchi Fang and Bing Wang. Mini-Splatting: Representing scenes with a constrained number of Gaussians. In *ECCV*, 2024.

Peter Fasogbon, Ugurcan Budak, Patrice Rondao Alface, and Hamed Rezazadegan Tavakoli. Camera-agnostic pruning of 3D Gaussian splats via descriptor-based beta evidence. In *BMVC*, 2026.

Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pp. 209–216, 1997.

Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. EAGLES: Efficient accelerated 3D Gaussians with lightweight EncodingS. In *ECCV*, 2024.

Alex Hanson, Allen Tu, Geng Lin, Vasu Singla, Matthias Zwicker, and Tom Goldstein. Speedy-Splat: Fast 3D Gaussian Splatting with sparse pixels and sparse primitives. In *CVPR*, 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 *CVPR*, 2025b.

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*, 37(6), 2018.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023.

Bernhard Kerbl, Andreas Meuleman, Georgios Kopanas, Michael Wimmer, Alexandre Lanvin, and George Drettakis. A hierarchical 3D Gaussian representation for real-time rendering of very large datasets. *ACM Transactions on Graphics*, 43(4), 2024.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4), 2017.

Itai Lang, Asaf Manor, and Shai Avidan. SampleNet: Differentiable point cloud sampling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7578–7588, 2020.

Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3D Gaussian representation for radiance field. In *CVPR*, 2024.

Bingxian Li, Yilong Li, Jingliang Peng, Peng-Shuai Wang, Fei Zhu, Guozheng Li, Chi Harold Liu, Guoping Wang, and Bo Pang. CVT-GS: Learning to simplify 3D Gaussian Splatting with Centroidal Voronoi Tessellation. *arXiv preprint arXiv:2609.08730*, 2026.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, Xuanmao Li, Xingpeng Sun, Rohan Ashok, Aniruddha Mukherjee, Hao Kang, Xiangrui Kong, Gang Hua, Tianyi Zhang, Bedrich Benes, and Aniket Bera. DL3DV-10K: A large-scale scene dataset for deep learning-based 3D vision. In *CVPR*, 2024.

Yifei Liu, Zhihang Zhong, Yifan Zhan, Sheng Xu, and Xiao Sun. MaskGaussian: Adaptive 3D Gaussian representation from probabilistic masks. In *CVPR*, 2025.

Saswat Subhajyoti Mallick, Rahul Goel, Bernhard Kerbl, Francisco Vicente Carrasco, Markus Steinberger, and Fernando de la Torre. Taming 3DGS: High-quality radiance fields with limited resources. In *SIGGRAPH Asia*, 2024.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

Panagiotis Papantonakis, Georgios Kopanas, Bernhard Kerbl, Alexandre Lanvin, and George Drettakis. Reducing the memory footprint of 3D Gaussian Splatting. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 7(1), 2024.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

Rolandos Alexandros Potamias, Giorgos Bouritsas, and Stefanos Zafeiriou. Revisiting point cloud simplification: A learnable feature preserving approach. In *European Conference on Computer Vision (ECCV)*, 2022.

Kerui Ren, Lihan Jiang, Tao Lu, Mulin Yu, Linning Xu, Zhangkai Ni, and Bo Dai. Octree-GS: Towards consistent real-time rendering with LOD-structured 3D Gaussians. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 627–635, 2011.

Andrew R. Runnalls. Kullback–Leibler approach to Gaussian mixture reduction. *IEEE Transactions on Aerospace and Electronic Systems*, 43(3):989–999, 2007.

D. J. Salmond, D. P. Atherton, and J. A. Bather. Mixture reduction algorithms for uncertain tracking. *IFAC Proceedings Volumes*, 21(9):775–780, 1988.

Tao Wang, Mengyu Li, Geduo Zeng, Cheng Meng, and Qiong Zhang. Gaussian herding across pens: An optimal transport perspective on global Gaussian reduction for 3DGS. In *NeurIPS*, 2025.

Jason L. Williams and Peter S. Maybeck. Cost-function-based hypothesis control techniques for multiple hypothesis tracking. *Mathematical and Computer Modelling*, 43(9–10):976–989, 2006.

Butian Xiong, Rong Liu, Tiantian Zhou, Meida Chen, Zhiwen Fan, and Andrew Feng. NanoGS: Training-free Gaussian splat simplification. In *ECCV*, 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 *CVPR*, 2026.

Yangming Zhang, Wenqi Jia, Wei Niu, and Miao Yin. GaussianSpa: An "optimizing-sparsifying" simplification framework for compact and high-quality 3D Gaussian Splatting. In *CVPR*, 2025.

A Additional Related Work

Learned point-cloud sampling and simplification networks, trained on a category or dataset, select task-relevant points (Dovrat et al., 2019; Lang et al., 2020) and can also refine the positions of selected representatives (Potamias et al., 2022); they operate on point geometry and are not supervised by the rendering of splatted primitives. More broadly, learned scores can guide greedy decisions in combinatorial procedures (Dai et al., 2017), and training on states visited by the learned policy itself can mitigate compounding errors (Ross et al., 2011). DeltaMerger learns such a score for recursive merging, and collects intervention labels along both analytic and predictor-guided trajectories (Appendix B.5).

B Implementation and Evaluation Details

B.1 Recursive Simplification Algorithm

Algorithm 1 Recursive pairwise simplification

Require: Gaussian representation $\mathcal{G}$, target count $1 \leq B \leq |\mathcal{G}|$, cost c, parent operator $\mathcal{M}$
Ensure: Simplified Gaussian representation containing B primitives
1: **while** $|\mathcal{G}| > B$ **do**
2: Construct the local candidate set $\mathcal{E}$ from $\mathcal{G}$.
3: Compute $c_e = c(\mathcal{G}, e; B)$ for each $e \in \mathcal{E}$.
4: Greedily select at most $|\mathcal{G}| - B$ disjoint pairs, prioritizing lower costs.
5: Replace the selected pairs in $\mathcal{G}$ with their analytic parents using $\mathcal{M}$.
6: **end while**
7: **return** $\mathcal{G}$

B.2 Implementation Details

Training data. We use 49,152 interventions from 64 DL3DV scenes (Ling et al., 2024): 2,048 state–budget groups of 24 candidates, each evaluated independently over eight views. Labels are collected along trajectories of the analytic NanoGS criterion and of an earlier frozen predictor, and the final predictor is trained on them (Appendix B.5).

Predictor and optimization. The 35-input MLP has hidden widths $64, 32$ (4,417 parameters), SiLU activations, and a linear output. We use 200 full-batch AdamW updates ($\mathrm{lr} = 10^{-3}$, weight decay 10^{-4}, gradient-norm clip 5), Huber threshold 1, and $\lambda_{\mathrm{rank}} = 4$, with three independent training seeds. The budget schedule is $\lambda = 1, 1, 0, 0, 0$ at the training retentions $0.1\%, 1\%, 5\%, 10\%, 25\%$. Inputs are normalized with training-only statistics (Appendix B.6), and the target scale also uses training-only statistics (Appendix B.7). Weights and normalization are frozen during evaluation.

Recursive simplification. We recompute four candidate neighbors and eight context neighbors per pass. Prefiltering removes opacities below $\min(0.1, \mathrm{median}_i\, \alpha_i)$ unless this would undershoot B, which is defined before filtering. We retain degree-three SH and use native NanoGS moment matching for every recursive parent; terminal refinement follows this stage (Appendix B.8).

B.3 Analytic Parent

We use the native parent operator of NanoGS (Xiong et al., 2026). Write $\mathbf{\Sigma}_k = \mathbf{R}_k\, \mathrm{diag}(\mathbf{s}_k^2)\mathbf{R}_k^\top$, $m_k = (2\pi)^{3/2}\alpha_k \prod_d s_{k,d}$, and $\omega_k = (m_k + 10^{-12})/(m_i + m_j + 2 \cdot 10^{-12})$ for $k \in \{i, j\}$. The parent parameters are

$$\boldsymbol{\mu}_p = \sum_k \omega_k \boldsymbol{\mu}_k, \qquad \alpha_p = \alpha_i + \alpha_j - \alpha_i\alpha_j, \qquad \mathbf{h}_p = \sum_k \omega_k \mathbf{h}_k, \tag{9}$$

$$\mathbf{\Sigma}_p = \sum_k \omega_k \left[\mathbf{\Sigma}_k + (\boldsymbol{\mu}_k - \boldsymbol{\mu}_p)(\boldsymbol{\mu}_k - \boldsymbol{\mu}_p)^\top\right] + 10^{-8}\mathbf{I}. \tag{10}$$

We symmetrize the covariance, floor its eigenvalues at 10^{-18}, and recover scales and rotation by eigendecomposition. The represented source-primitive count is additive, whereas opacity composition need not conserve integrated mass. This parent defines training interventions, parent descriptors, and deployed replacements during recursive simplification.

B.4 Interpreting the Rendered-Intervention Target

In Equation 6, the immediate change $\mathcal{D}_{t,e} \geq 0$ measures how far a merge moves the current rendering, while the cross term $\mathcal{K}_{t,e}$ is signed: positive when the perturbation adds to the accumulated error and negative when it opposes it. Such repairs are frequent in our training labels: at every requested budget, about 30% of sampled candidates reduce $\mathcal{E}^0$, and the median ratio $|\mathcal{K}_{t,e}|/\mathcal{D}_{t,e}$ lies between 3.2 and 3.6. The sign of analytic costs is not what limits them, since greedy selection uses only their order; rather, they compare children with their parent and cannot observe the accumulated error $\boldsymbol{\varepsilon}_t$ on which $\mathcal{K}_{t,e}$ depends. The learned descriptors need not determine $\boldsymbol{\varepsilon}_t$ either: the predictor only has to rank candidates. For $\lambda > 0$, the target of Equation 7 ranks candidates like $\Delta\mathcal{E}^0_{t,e} + \tau\,\mathcal{D}_{t,e}$ with $\tau = (1-\lambda)/\lambda$: fidelity to the original rendering with a proximal penalty on changes to the current state. This describes the target; the network is trained only at the five budgets, where $\lambda \in \{0, 1\}$.

B.5 Training-State and Intervention Sampling

Each of the 64 training scenes contributes 16 state groups from each of two fixed acquisition policies: the analytic NanoGS criterion and an earlier frozen predictor, distinct from the final model. Visited states are stratified into retention intervals $(25\%, 100\%]$, $(10\%, 25\%]$, $(5\%, 10\%]$, $(1\%, 5\%]$, and $(0, 1\%]$; sampling keeps the last eligible state for each budget and policy, and an early state for trajectories targeting 0.1% and 1%. Within a group, we request six accepted pairs, six alternatives conflicting with accepted pairs (prioritized by their cost gap to the accepted pair), and six pairs with the largest absolute difference between analytic and learned ranks; after deduplication, uniform samples from the candidate graph fill the group to 24 interventions. No rendered label is

used for candidate sampling. A separate uniform reservoir of 512 pairs per group provides feature-normalization statistics.

All 24 interventions start from the same unmodified state and replace only their own pair. Supervision uses eight cameras from a fixed 16-view panel selected for pose coverage, disjoint from held-out evaluation views. We render full images with degree-three SH and a black background, clip RGB to $[0, 1]$, and pool squared differences over all pixels and channels, so views contribute in proportion to their pixel counts.

B.6 Selected 35-Input Representation

Pair and neighborhood descriptors (23). Each primitive has nine scalar descriptors: log opacity; log scale product; log sum of pairwise scale products; log represented source count; mean RGB SH DC coefficient; a log directional SH-norm summary (degrees one and two for source primitives, one to three for merged parents, identically in training and deployment); log mean distance to eight neighboring centers; DC coherence $1/(1 + \|\mathbf{h}_i^{(0)} - \overline{\mathbf{h}}_{\mathcal{N}_i}^{(0)}\|_2)$ with these neighbors, where the bar denotes the unweighted mean over the eight neighbors; and log sum of their opacities. Pairwise sums and absolute differences of these nine scalars give 18 inputs. Five further inputs are log center distance normalized by the current median nearest-neighbor distance, absolute principal-axis alignment, absolute difference in log maximum scale, log Gaussian overlap, and squared distance between all 48 SH coefficients.

Directional appearance (3). For RGB SH vectors $\mathbf{a}_\ell, \mathbf{b}_\ell$ of the two children, we use $u_\ell = \log(\|\mathbf{a}_\ell\|_2^2 + \|\mathbf{b}_\ell\|_2^2 + 10^{-12})$ for $\ell = 1, 2, 3$.

Pair mass and proposed parent (5). Using the integrated masses above, set $L = \log(m_i + m_j)$, $\omega_k = m_k/(m_i + m_j)$ (the mass shares of Appendix B.3, without the 10^{-12} stabilizer), $a_k = \log \max_d s_{k,d}$, and $b_k = \log(\max_d s_{k,d}/\min_d s_{k,d})$. The five inputs are

$$\left[L,\ |\log m_i - \log m_j|,\ \log m_p - L,\ a_p - \sum_k \omega_k a_k,\ b_p - \sum_k \omega_k b_k \right]. \tag{11}$$

The remaining four inputs are the state and budget coordinates $[r_t, \log r_t, \log q, \log \rho]$ defined in Section 3.2.1.

Preprocessing. With the median η_j and interquartile range ν_j of valid values in the training reservoirs, we encode $z_j = \mathrm{clip}((x_j - \eta_j)/\max(\nu_j, 10^{-6}), -4, 4)$, replacing invalid values (including mass and parent descriptors with nonpositive masses or scales) by η_j; no validity flags are appended. Pairs are put in canonical index order and scored once.

Input ablations. Removing mass/parent (30 inputs), budget (34), or population-state (32) features tests conditional contributions, since correlated cues remain (Table 4). Removing the mass/parent block causes the largest losses, 0.20–0.34 dB at 5%–25% retention, and removing the requested budget costs 0.05–0.08 dB from 0.1% to 5%. Removing the population-state block changes PSNR by at most 0.04 dB; these inputs were retained from the development selection. Adding the NanoGS geometric cost as a 36th input gains 0.06 dB at 0.1% and changes PSNR by at most 0.02 dB elsewhere, so adding it gives no consistent improvement in this ablation.

B.7 Target Scaling and Training Weights

Shared target scale. Targets are normalized as $y = Y/s$ by a training-only scalar. Besides the rendered components of Y, the training cache records $\Delta\mathcal{E}_{t,e}^{\mathrm{GT}} = \mathrm{MSE}(\mathbf{I}_{t,e}, \mathbf{J}) - \mathrm{MSE}(\mathbf{I}_t, \mathbf{J})$ against training photographs $\mathbf{J}$, over the same views and pixels. With $\mathcal{U}$ the set of $\mathcal{D}_{t,e}$ and $|\Delta\mathcal{E}_{t,e}^{\mathrm{GT}}|$ over all training interventions,

$$s = \max\left(10^{-12}, Q_{0.75}(\{u \in \mathcal{U} : u > 0\})\right) \approx 1.011 \times 10^{-7}. \tag{12}$$

Photographs enter the merge-cost predictor only through this scalar, not as a prediction target (the terminal refiner, by contrast, is trained directly against them). Deployment ranks the normalized outputs and does not require s.

Table 4: Input-descriptor ablations, complementing Table 3. PSNR difference (dB) from the selected predictor (35 inputs, $\lambda = 1, 1, 0, 0, 0$), averaged over three training seeds and the 46 evaluation scenes, without terminal refinement. Positive values favor the variant. Input counts are in parentheses. The budget schedule itself was selected on a held-out validation set (Appendix B.7).

Variant	0.1%	1%	5%	10%	25%
Input descriptors					
Without mass/parent block (30)	-0.044	-0.053	-0.204	-0.277	-0.335
Without requested budget (34)	-0.047	-0.077	-0.061	-0.023	-0.003
Without state block (32)	-0.026	-0.003	-0.007	+0.023	+0.036
With NanoGS geometry (36)	+0.059	-0.004	-0.004	-0.015	-0.020

Regression and ranking. For candidates $\mathcal{Q}_g$ in group g, let $\mathcal{P}_g = \{(a, b) : a < b,\ y_{g,a} \neq y_{g,b}\}$ and $\tau_{ab} = \operatorname{sign}(y_{g,b} - y_{g,a})$. The terms in Equation 8 are

$$\mathcal{L}^{(g)}_{\text{reg}} = \frac{1}{|\mathcal{Q}_g|} \sum_{a \in \mathcal{Q}_g} \text{Huber}(\hat{y}_{g,a} - y_{g,a}), \tag{13}$$

$$\mathcal{L}^{(g)}_{\text{rank}} = \frac{1}{\max(1, |\mathcal{P}_g|)} \sum_{(a,b) \in \mathcal{P}_g} \beta_{g,ab} \operatorname{softplus}[-\tau_{ab}(\hat{y}_{g,b} - \hat{y}_{g,a})]\,, \tag{14}$$

with the settings of Appendix B.2. The mixed target and linear network output are signed and are not clipped.

Comparison weights. Let $d_{ab} = |\mathcal{D}_{g,a} - \mathcal{D}_{g,b}|$, and let q_g be the 75th percentile of its positive values. We define $h_{g,ab} = 1 + 3\min(d_{ab}/q_g, 1)$ and divide by the mean of h over positive-gap comparisons to obtain $\gamma_{g,ab}$. Tied comparisons use $h = 1$ and the same normalizer; if every gap is zero, set $\gamma = 1$. With $\lambda_g = \lambda(\rho_g)$ for the budget ρ_g of group g and $0 < \lambda_g < 1$, normalize $b_{g,ab} = (1 - \lambda_g)\gamma_{g,ab} + \lambda_g$ to unit mean over $\mathcal{P}_g$, giving $\beta_{g,ab}$. At the endpoints, use $\beta = \gamma$ for $\lambda_g = 0$ and $\beta = 1$ for $\lambda_g = 1$. Groups with no informative comparison have zero ranking loss. Thus mixed targets determine comparison directions, while $\mathcal{D}$ gaps determine the nonuniform part of their weights.

Group weights. For S training scenes, B_s represented budgets per scene, $P_{s,b}$ acquisition policies per scene–budget cell, and $G_{s,b,p}$ groups per cell, we set $w_g \propto (SB_sP_{s,b}G_{s,b,p})^{-1}$ and normalize so that $\sum_g w_g = 1$. This balances the nested sampling levels despite unequal collection counts.

Hyperparameter selection. The budget schedule λ, the predictor width, and the number of updates were selected on a held-out validation set of 10 DL3DV scenes, disjoint from the training and evaluation scenes.

B.8 Terminal Refinement Details

Corrections. For each final representative r, a 40-dimensional descriptor $\mathbf{x}_r$ summarizes its attributes, the source primitives of its group (those retained after prefiltering), and its eight nearest final neighbors. Descriptors are computed once on the unrefined state, so the pipeline must keep source-group membership or its sufficient statistics; a standalone compressed PLY need not contain this information. A budget-conditioned network predicts bounded actions $\mathbf{a}_r \in [-1, 1]^7$, applied as

$$\begin{aligned} \mathbf{h}^{(0)\prime}_r &= \mathbf{h}^{(0)}_r + \mathbf{a}^{\text{DC}}_r, \qquad \alpha'_r = \sigma(\operatorname{logit}\alpha_r + 2.5a^{\alpha}_r)\,, \\ s'_{r,d} &= s_{r,d}\exp(0.75a^{s}_{r,d}), \qquad d \in \{1, 2, 3\}. \end{aligned} \tag{15}$$

The DC bound is in SH-coefficient units; the opacity update preserves the endpoints 0 and 1, and zero actions give the identity. Singletons receive zero actions.

Descriptor. Mass is proportional to $\alpha \prod_d s_d$, and the group extent is the maximum member–parent center distance plus the maximum member scale-vector norm, floored at 10^{-4}. *Base (21;*

unweighted member moments): log-base-two group cardinality; log extent; log total member mass; parent opacity logit; three parent log scales; log scale ratios s_x/s_z and s_y/s_z; member positional RMS about the parent divided by extent; member opacity mean and standard deviation; three parent SH-DC coefficients; three member SH-DC standard deviations; and parent SH RMS for bands one to three. *Mass-weighted group (16):* with normalized member masses as weights (uniform if the total mass is at most 10^{-12}), log entropy-effective member count; largest and second-largest mass shares; spatial RMS about the weighted centroid divided by extent; log condition number of the member-center covariance; radial fourth moment divided by squared second moment; maximum member–parent and parent–weighted-centroid distances divided by extent; three parent log scales minus weighted member log scales; member log-scale RMS; parent minus weighted member log volume and log anisotropy; log parent-to-group mass ratio; and one minus effective count divided by cardinality. *Neighborhood (3):* with $\mathcal{N}_r$ the $\min(8, B-1)$ nearest other final centers,

$$\left[\log\max\left(\frac{\overline{d}_{r,\mathcal{N}_r}}{\max_d s_{r,d}}, 10^{-12}\right), \quad \frac{1}{1+\|\mathbf{h}_r^{(0)} - \overline{\mathbf{h}}_{\mathcal{N}_r}^{(0)}\|_2}, \quad \log\left(\sum_{k\in\mathcal{N}_r}\alpha_k + 10^{-12}\right)\right], \tag{16}$$

where bars denote unweighted means over $\mathcal{N}_r$ of center distances and SH-DC coefficients; all three are zero if there is no neighbor. No analytic merge cost is a refiner input. This descriptor was chosen, in an earlier development campaign, among 37-, 40-, and 41-input variants on four DL3DV development scenes disjoint from the training and evaluation scenes.

Architecture. The refiner (7,751 parameters) consists of an affine $40 \to 64$ projection followed by feature-wise linear modulation (FiLM; Perez et al., 2018) and SiLU, one residual block $\mathbf{h} + \mathrm{SiLU}(\mathrm{FiLM}_2(W\mathbf{h}+\mathbf{b}, \widetilde{\rho}))$, and a zero-initialized $64 \to 7$ affine head followed by $\tanh$. Each FiLM module maps $\widetilde{\rho}$, the training-standardized $\log\rho$, to 64 scales and shifts applied as $(1+\boldsymbol{\gamma}) \odot \mathbf{h} + \boldsymbol{\beta}$. Features are standardized with training means and standard deviations (each scene–budget cell weighted equally, representatives equally within a cell), floored at 10^{-6} and clipped to $[-8, 8]$.

Training. Refiners are trained on the same 64 DL3DV training scenes as the merge-cost predictor, at the five retentions (320 scene–budget cells), with the planner frozen: one refiner per predictor seed and, for Section 4.3, three after analytic selection, with the same scenes, retentions, and views. No evaluation scene is used. With descriptors and groups fixed, we apply the actions to every merged representative of a training state $\mathcal{G}_{s,b}$, render the corrected scene $\widetilde{\mathcal{G}}_{s,b}(\phi)$ from a FIT view v, and minimize its RGB error against the photograph $\mathbf{J}_v$,

$$\mathcal{L}_{\mathrm{ref}}(\phi) = \mathbb{E}_{s,b,v}\left[\frac{1}{3H_vW_v}\left\|\mathcal{R}\big(\widetilde{\mathcal{G}}_{s,b}(\phi), v\big) - \mathbf{J}_v\right\|_F^2\right], \tag{17}$$

with s, b, and v drawn uniformly; gradients flow through the differentiable renderer, and no per-cell action label is used. Each scene has 16 FIT views and four SELECT views, disjoint and reused across budgets, taken at indices $\min(N_v-1, \lfloor (i+p)N_v/8 \rfloor)$, $i = 0, \ldots, 7$, in camera-file order: FIT uses $p = 0.25$ and 0.75, SELECT alternate indices at $p = 0.5$. We use 6,000 AdamW updates of four sampled renderings each, learning rate 10^{-3} (100-update linear warm-up, cosine decay to zero), weight decay 10^{-4}, and gradient-norm limit 1. An exponential moving average of the weights (decay 0.99) is evaluated every 500 updates on the SELECT views of all 320 cells, together with the identity, and the lowest mean MSE is kept; all six refiners select update 5,500 or 6,000.

Deployment scope and parity. The refiner predicts once on the terminal state, and the output is a standard Gaussian asset rendered without any network; since scales and opacities change, the fixed count does not by itself guarantee identical rendering cost. The timings of Appendix B.12 include descriptor construction. Because the refiner is trained against photographs, its gains may correct source-reconstruction errors as well as merging errors; training against original renders instead is not evaluated. Before refinement, every test state of both planners reproduces its merge-only evaluation to within 0.008 dB PSNR (near-ties can resolve differently across CPU families), and each paired comparison uses states planned once and shared by all its arms.

B.9 Evaluation Protocol and Additional Results

The evaluation uses 46 scenes: ten DL3DV, nine Deep Blending, nine Mip-NeRF 360, eight NeRF Synthetic, and ten Tanks and Temples. They are disjoint from the 64 training scenes, but some were

inspected during method development, so this is not an untouched test set. DL3DV sources are the pretrained 3DGS models distributed with DL3DV (Ling et al., 2024); the others were reconstructed with the official 3DGS implementation (Kerbl et al., 2023) (30,000 iterations, SH degree 3). Table 6 lists every scene with its budget-averaged results, and Table 7 the training scenes.

Metrics use the full held-out views of each source's train/test split: PSNR is averaged over views, SSIM uses an 11-pixel window, and LPIPS uses VGG. All methods share reference images, cameras, renderer settings, and backgrounds. Requested counts refer to the original source before filtering: our pipeline uses $\max(1, \lfloor \rho N_0 \rfloor)$ and skips filtering if it would undershoot this count, while official NanoGS keeps its own filtering and ceiling convention and GHAP removes empty clusters (Appendix B.11), so table columns specify requested retentions. For refinement results (Figure 2, Table 2, and Tables 11 and 12), each of the three predictor seeds is compared before and after its own refiner; we average views within a scene–budget cell, then seeds, then cells, and both planners use the same cells. Table 5 reports alternative aggregation and individual training replicates.

Aggregation and training replicates. The main aggregation gives equal weight to physical scenes. For comparison, Table 5 first averages scenes within each dataset and then gives the datasets equal weight. Seed variability describes repeated training, not uncertainty over the population of scenes.

Table 5: PSNR gains over NanoGS ($k = 4$): aggregation conventions and individual training replicates.

Aggregation	0.1%	1%	5%	10%	25%
Equal scene weights	+0.148	+0.192	+0.440	+0.524	+0.465
Equal dataset weights	+0.164	+0.191	+0.429	+0.516	+0.462
Training seed 0	+0.090	+0.150	+0.423	+0.505	+0.403
Training seed 1	+0.142	+0.210	+0.431	+0.534	+0.498
Training seed 2	+0.213	+0.215	+0.466	+0.533	+0.495

B.10 Failure Cases

Losses to NanoGS concentrate at the most aggressive budgets: without refinement, the learned cost is worse on 13 of 46 scenes at 0.1% and on 9 at 1%, but on at most four from 5% upward. At 0.1%, the largest losses occur on Tanks and Temples (Family -0.65 dB, Ignatius -0.50 dB) and on two DL3DV scenes (-0.52 and -0.49 dB), and the DL3DV mean is slightly negative (-0.04 dB). At 1%, the largest losses are on Mip-NeRF 360 Kitchen (-0.62 dB) and one DL3DV scene (-0.32 dB); Tanks and Temples Ignatius is the only scene that still loses at 10%. Against NanoGS with $k = 16$, the only budget with a mean loss is 0.1% (-0.03 dB). With terminal refinement, analytic selection followed by its own refiners is better at 0.1% on 37 of 46 scenes (Section 4.3). Refinement itself lowers PSNR on 8 of the 230 cells: all five budgets of one DL3DV scene (`063d0a7a6f`, about 0.8 dB on average, where the refiners trained after analytic selection also lose), NeRF Synthetic Ficus at 0.1% and 25%, and Tanks and Temples Horse at 0.1%.

B.11 GHAP without Photometric Fine-Tuning

GHAP (no FT) runs only the released geometric reduction (commit `27b3a581`; seed 42, tolerance 10^{-5}, at most 100 iterations per leaf) on each source PLY, without the subsequent photometric recovery of color and opacity; it uses no images or learned model, and does not represent the full published method. For small budgets, we cap the KD-tree depth at $d = \min(10, \lfloor \log_2 B \rfloor)$, and a leaf whose float32 solve raises an objective-increase error is retried in float64 with unchanged settings. Because GHAP removes empty clusters, exported counts fall below the request by less than 0.223%. Evaluation uses the same test views and metrics as for the other methods.

B.12 Computational Cost

Timing protocol. All end-to-end timings run one process at a time on a dedicated host (AMD Ryzen Threadripper 7960X, 24 cores). Each method runs as a complete PLY-to-PLY process (start-

Table 6: Per-scene PSNR (dB) averaged over the five budgets. DeltaMerger entries average the three predictor seeds; refined entries use each planner's own refiners. DL3DV scenes are identified by the first ten characters of their hash.

Scene	NanoGS k=4	NanoGS k=16	DeltaMerger w/o ref.	Δ vs. k=4	Analytic + ref.	DeltaMerger
DL3DV						
063d0a7a6f	27.81	27.32	28.59	+0.78	27.41	27.79
0a1860d3a6	18.71	18.29	19.02	+0.31	19.58	19.72
27e1f6dc8b	21.95	21.66	22.52	+0.56	23.05	23.39
3a17b1db0f	25.92	25.63	26.40	+0.48	27.46	27.86
3bf90a4efe	18.82	18.41	19.02	+0.21	19.42	19.45
427d58201f	23.04	22.83	23.78	+0.74	24.86	25.01
517bab668e	22.19	22.08	22.52	+0.33	23.35	23.43
86b072eafd	17.82	17.54	18.08	+0.26	18.84	18.98
8ef70669dd	26.04	24.87	26.46	+0.42	27.10	27.41
c45b0de652	21.42	21.49	22.00	+0.58	22.77	22.83
Deep Blending						
bedroom	22.48	23.40	23.77	+1.29	23.91	24.45
boats	19.92	19.83	20.57	+0.65	20.82	21.16
creepyattic	25.19	25.19	25.59	+0.41	26.08	26.26
drjohnson	23.77	23.66	24.11	+0.35	24.89	24.95
library	20.42	20.13	20.32	−0.10	21.61	21.64
museum-2	20.91	20.44	21.24	+0.33	21.70	21.83
nightsnow	23.68	23.11	24.06	+0.38	24.69	24.89
playroom	24.52	24.77	24.94	+0.42	25.63	25.75
saintanne	20.40	20.04	20.68	+0.29	20.88	21.14
Mip-NeRF 360						
bicycle	19.73	19.46	20.02	+0.29	20.52	20.67
bonsai	22.73	22.62	22.90	+0.18	23.49	23.47
counter	21.53	21.67	22.00	+0.47	22.52	22.62
flowers	17.25	16.94	17.71	+0.45	17.90	18.11
garden	20.18	20.23	20.63	+0.45	21.31	21.47
kitchen	21.86	21.96	21.76	−0.10	22.94	22.86
room	24.08	24.05	24.37	+0.29	24.80	24.94
stump	20.89	20.64	21.18	+0.29	21.58	21.77
treehill	18.77	18.55	19.09	+0.32	19.51	19.64
NeRF Synthetic						
chair	25.27	25.42	25.52	+0.26	26.05	26.13
drums	21.83	21.86	21.99	+0.16	22.53	22.44
ficus	26.99	26.60	27.42	+0.43	26.98	27.46
hotdog	26.63	27.19	27.22	+0.59	27.77	27.97
lego	23.48	23.80	23.84	+0.35	25.01	24.92
materials	23.14	23.26	23.48	+0.34	23.65	23.79
mic	26.65	27.06	26.86	+0.21	28.06	28.12
ship	22.30	22.29	22.36	+0.06	23.16	23.22
Tanks and Temples						
barn	19.39	19.34	19.88	+0.49	20.37	20.54
caterpillar	16.50	16.12	16.60	+0.09	17.53	17.40
courthouse	17.38	17.13	17.68	+0.30	18.40	18.48
family	18.29	18.27	18.49	+0.20	19.39	19.46
horse	17.09	17.08	17.55	+0.46	18.11	18.42
ignatius	16.61	16.22	16.50	−0.11	17.98	18.01
lighthouse	16.76	16.79	16.89	+0.13	17.70	17.75
meetingroom	19.55	19.40	19.77	+0.23	20.51	20.65
train	15.91	15.63	15.98	+0.06	16.89	16.86
truck	17.94	17.57	18.65	+0.72	19.79	20.03

Table 7: The 64 DL3DV training scenes, identified by the first ten characters of their hash. The 46 evaluation scenes are listed in Table 6.

03a68164db	04070013a0	0510d00961	05500846b3	075871bafd	0777b243c0	0ef7c0d6e3	10dc690f34
1892709c7d	1a6e5c0262	1c276f55a2	1f7b40e070	24c54fbcf8	25c3d73186	25f18701c0	291292e899
3c05f8461e	4299796cbd	463cad3b69	476278190c	49a22b523a	4df8b9df7c	4e49fea200	52bb81432d
5b1c6a4509	5d6481db3d	5d97f533bd	61e7614b68	6780efcf94	6a915b1e3b	6cb22a1620	6d22162561
6de1b86100	6efc000080	7cc121c8b2	80bbaa1bbd	812232578f	825109b6af	874a954cfd	8aeea98048
8b3a6cd126	8cebb92724	90c06ab59d	911435f1d9	952cb5fc91	9d1913c610	a62c330f54	af2818905a
b0d21741fb	bd130a4757	bdbde76ae5	c7576be88b	c9fb41b9d3	cbaac3e651	ceb252f5d4	d1797f89b3
d591f08210	d5fb5a2c85	dcdb59c80b	e70bc867c7	e7af8cf57f	e822c320a8	f43a36bd40	fe623fef3c

up, imports, model loading, reading and writing the PLY), pinned to the same four physical cores with idle SMT siblings and BLAS limited to four threads; the source PLY is read once before each cell, methods run in rotating order per cell, and peak resident memory is taken from the process accounting. NanoGS uses its official optimized CPU branch, whose outputs are identical to its main branch; timing NanoGS ($k = 4$) again once per scene after all other methods changes its time by 0.4% (geometric mean over 46 cells).

Optimized CPU implementation. The reference implementation used for all other quality results favors auditability over speed. The optimized implementation of the same pipeline uses parallel compiled kernels for candidate search, descriptors, scoring, selection, and parent updates, evaluates the MLP inside the descriptor loop, releases the source scene after prefiltering, and applies the refiner with a compiled kernel. Compiled kernels can differ from the reference in the last bits, so near-ties of the greedy selection can resolve differently. Table 8 and Figure 4 therefore report the quality of the timed outputs themselves, with seed-0 models: every output was regenerated with the timed options, verified to be byte-identical to the timed one, and evaluated with the protocol of Appendix B.9. Relative to the reference evaluation of the same models, mean PSNR differs by -0.001 to $+0.019$ dB across the four configurations (mean absolute per-cell difference at most 0.085 dB). The learned rows are 0.03 dB below the three-seed means of Table 2 because seed 0 is slightly below average. Our pipeline shares PLY reading, candidate search, and moment matching with NanoGS and implements them more efficiently; the ratio to analytic selection compares the same pipeline with learned or analytic selection, whose decisions and trajectories differ.

Table 8: End-to-end cost on all 46 evaluation scenes and five budgets (230 cells per method), each method a complete PLY-to-PLY process on the same four pinned cores of one dedicated host, one process at a time. Time is the arithmetic mean over cells, dominated by large scenes; ratios are geometric means over cells of per-cell time ratios, to NanoGS ($k = 4$) and, for our learned cost, to analytic selection in the same optimized pipeline (with refinement: to analytic selection followed by its own refiner). RAM is the mean peak resident memory. Our rows use the optimized CPU build with seed-0 models, and their PSNR is measured on the timed outputs themselves; NanoGS PSNR is the reference evaluation of Table 1.

Method	Time (s)	vs. NanoGS	vs. analytic	RAM (GB)	PSNR (dB)
NanoGS ($k = 4$)	12.0	1.00	–	1.5	21.39
NanoGS ($k = 16$)	35.8	2.91	–	1.5	21.26
Analytic selection (ours)	5.4	0.47	1.00	1.4	21.39
Analytic selection + refiner	5.5	0.49	1.00	1.4	22.38
DeltaMerger w/o refinement	6.8	0.59	1.25	1.5	21.71
DeltaMerger	7.0	0.60	1.24	1.5	22.47

Rendering and offline cost. Table 9 reports rendering speed and file size, and Table 10 the offline training cost; the hardware of each measurement is stated in the tables.

B.13 Additional refinement results

Tables 11 and 12 report paired refinement deltas by retention and by dataset: each predictor seed's refined output is compared with its own unrefined states under the same views, and deltas are averaged over the three seeds. Over the 230 cells, mean PSNR changes from 21.739 to 22.503 dB, SSIM

Table 9: Rendering speed (milliseconds per frame over all test views, one RTX 4090) and PLY size of the original, simplified (DeltaMerger w/o refinement, seed 0), and refined (DeltaMerger) assets on the same scenes. The original asset is the same in every row; its timings differ only by measurement variability. Refinement keeps the primitive count; its rendering time, averaged over scenes, differs by less than 5% at every retention.

Retention	Original	Simplified	Refined	Original	Simplified	Refined
		ms / frame			PLY (MB)	
0.1%	12.58	3.07	3.11	499.0	0.5	0.5
1%	12.96	3.53	3.66	499.0	5.0	5.0
5%	13.23	4.48	4.48	499.0	24.9	24.9
10%	12.69	5.56	5.75	499.0	49.9	49.9
25%	12.70	8.04	7.66	499.0	124.7	124.7

Table 10: Offline training cost. †Optimized collector: after-merge scenes are built and rendered on the GPU, eight views per call, with label statistics reduced on the GPU (on 96 interventions checked against the reference collector, all label statistics agree within 6×10^{-15} relative difference, $|\Delta| / \max(|\mathrm{ref}|, 10^{-12})$). Timed with the same protocol on six jobs from three training scenes (88–879 MB) and extrapolated linearly in scene size to all 128 jobs, run six at a time on four cores each; the GPU is active for about 4 min and the collection is CPU-bound (about 19 CPU core-hours). Training photographs for the refiner occupy 21 GB.

Stage	Hardware	Time	Storage
Label collection (49,152 interventions, 8 views)	1 RTX 5090 + 24-core CPU	≈48 min†	1.1 GB
Merge-cost predictor (200 updates, per seed)	1 RTX 4090	14 s	62 kB
Refiner training states (320 per predictor)	CPU (4 workers/job)	≈5 CPU-job-h	12.7 GB
Shared refiner (6,000 updates, per seed)	1 RTX 4090	≈15 min	38 kB

from 0.71532 to 0.72516, and LPIPS from 0.45953 to 0.44704; the seed-averaged change improves PSNR on 222 cells, SSIM on 217, and LPIPS on 206. NeRF Synthetic has a small mean LPIPS increase ($+0.0003$) despite positive PSNR and SSIM changes.

Table 13 adds SSIM and LPIPS for the four configurations of Table 2: with each planner followed by its own refiner, learned selection has better mean SSIM and LPIPS, but analytic selection is better on both at 0.1% and on LPIPS at 1%.

Table 11: Refinement deltas by retention on the 46 test scenes, paired against the same unrefined states and averaged over the three predictor seeds, each with its own refiner. The last two columns count cells whose PSNR improves or regresses.

Retention	n	ΔPSNR (dB)↑	ΔSSIM↑	ΔLPIPS↓	+	−
0.1%	46/46	+0.612	+0.00318	−0.00568	43	3
1%	46/46	+0.908	+0.00807	−0.01147	45	1
5%	46/46	+0.970	+0.01320	−0.01773	45	1
10%	46/46	+0.863	+0.01483	−0.01753	45	1
25%	46/46	+0.467	+0.00991	−0.01002	44	2
All	230/230	+0.764	+0.00984	−0.01249	222	8

Table 12: Refinement deltas by dataset over the five retentions, averaged over the three predictor seeds. Each scene–budget cell has equal weight; n counts evaluated cells.

Dataset	n	ΔPSNR (dB)↑	ΔSSIM↑	ΔLPIPS↓
DL3DV	50/50	+0.748	+0.01023	−0.01730
Deep Blending	45/45	+0.754	+0.00745	−0.01394
Mip-NeRF 360	45/45	+0.657	+0.01152	−0.01193
NeRF Synthetic	40/40	+0.669	+0.00444	+0.00032
Tanks and Temples	50/50	+0.961	+0.01441	−0.01712
All	230/230	+0.764	+0.00984	−0.01249

Table 13: PSNR, SSIM, and LPIPS of the four configurations of Table 2 on the same 230 cells (46 scenes), averaging seeds and then scenes.

Configuration	0.1%	1%	5%	10%	25%	Mean
PSNR ↑						
Analytic (A_0)	16.86	19.79	21.94	23.13	25.21	21.39
+ refinement (A_1)	17.79	20.77	23.06	24.21	25.96	22.36
Learned (L_0)	17.01	19.98	22.38	23.65	25.67	21.74
+ refinement (L_1)	17.62	20.89	23.35	24.52	26.14	22.50
SSIM ↑						
Analytic (A_0)	0.6238	0.6625	0.7122	0.7448	0.8034	0.7093
+ refinement (A_1)	0.6300	0.6717	0.7277	0.7641	0.8195	0.7226
Learned (L_0)	0.6249	0.6649	0.7182	0.7542	0.8143	0.7153
+ refinement (L_1)	0.6281	0.6730	0.7314	0.7690	0.8242	0.7252
LPIPS ↓						
Analytic (A_0)	0.5973	0.5273	0.4513	0.4101	0.3409	0.4654
+ refinement (A_1)	0.5896	0.5146	0.4303	0.3873	0.3233	0.4490
Learned (L_0)	0.5966	0.5278	0.4457	0.4004	0.3271	0.4595
+ refinement (L_1)	0.5909	0.5163	0.4280	0.3829	0.3171	0.4470

C ADDITIONAL QUALITATIVE RESULTS

We show eight additional scenes at all five retention budgets ($0.1\%, 1\%, 5\%, 10\%, 25\%$), for 40 scene–budget comparisons. The scenes cover Mip-NeRF 360 (Barron et al., 2022), Tanks and Temples (Knapitsch et al., 2017), Deep Blending (Hedman et al., 2018), NeRF Synthetic (Mildenhall et al., 2020), and DL3DV (Ling et al., 2024). All comparisons use the reference pipelines: the official NanoGS implementation with $k = 4$ (Xiong et al., 2026), the final predictor (seed 0) without refinement, and the same predictor with its paired shared refiner. No per-scene fine-tuning is applied.

Fixed, stratified view selection. We select views using existing held-out PSNR measurements at 5% retention only. The ranking quantity is the full-view PSNR gain of DeltaMerger without refinement over NanoGS ($k = 4$). For each dataset and budget separately, its empirical percentile is 100 times the fraction of paired test views with gain less than or equal to the selected view's gain. The pools include all matched views from the 46 primary evaluation scenes, with one vote per view; these percentiles therefore do not use the equal-scene weighting of our aggregate results.

For the six named scenes, we fix the scene and a lower-quartile, median, or upper-quartile target, then choose the view closest to that target percentile at 5% (Table 14). For DL3DV, we choose the closest view to the lower-quartile target first, then the closest view to the upper-quartile target from a different scene. Ties are broken lexicographically by scene and view identifier. This is a deliberately stratified illustration, not a random or unbiased sample of evaluation views. The selected examples include unfavorable cases: DeltaMerger w/o refinement has lower full-view PSNR than NanoGS ($k = 4$) in 11 of the 40 comparisons (Bonsai, Playroom, Train, Truck, and both DL3DV scenes), the complete DeltaMerger in 2, and refinement lowers PSNR in 2 (see also Appendix B.10). In particular, a target percentile does not require a positive gain. The other four budgets do not influence selection, and their percentiles need not match these targets.

Table 14: Fixed views used at every budget. Target and observed empirical percentiles refer only to the unrefined gain at 5% retention. DL3DV scene names are shown as hash prefixes.

Dataset	Scene	Test view	Target	Observed
Mip-NeRF 360	Garden	`DSC08124.JPG`	75	74.0
	Bonsai	`DSCF5669.JPG`	25	25.2
Tanks and Temples	Train	`00065.jpg`	25	24.9
	Truck	`000121.jpg`	75	74.9
Deep Blending	Playroom	`DSC05711.jpg`	50	49.8
NeRF Synthetic	Lego	`r_124`	50	49.9
DL3DV	`86b072ea`	`frame_00105`	25	25.1
	`517bab66`	`frame_00009`	75	75.2

Reading the comparisons. After selection, each scene keeps exactly the same test view and crop at all five budgets; no favorable-gain filter is applied at the other budgets. Crop locations are chosen from the reference images alone and shared by all methods. Figure 5 combines the 40 comparisons into a single figure continued across two pages. Each scene forms one band, with its reference shown once on the left and the five budgets in columns. Within each budget, rows N, U and D show NanoGS ($k = 4$), DeltaMerger without refinement and DeltaMerger, respectively. Each method row places the complete view beside its fixed crop; the number below is full-view PSNR, not crop PSNR. The images originate from full-resolution evaluation renders and are scaled for compact display; reported PSNR values are unchanged. Both favorable and unfavorable outcomes remain visible.

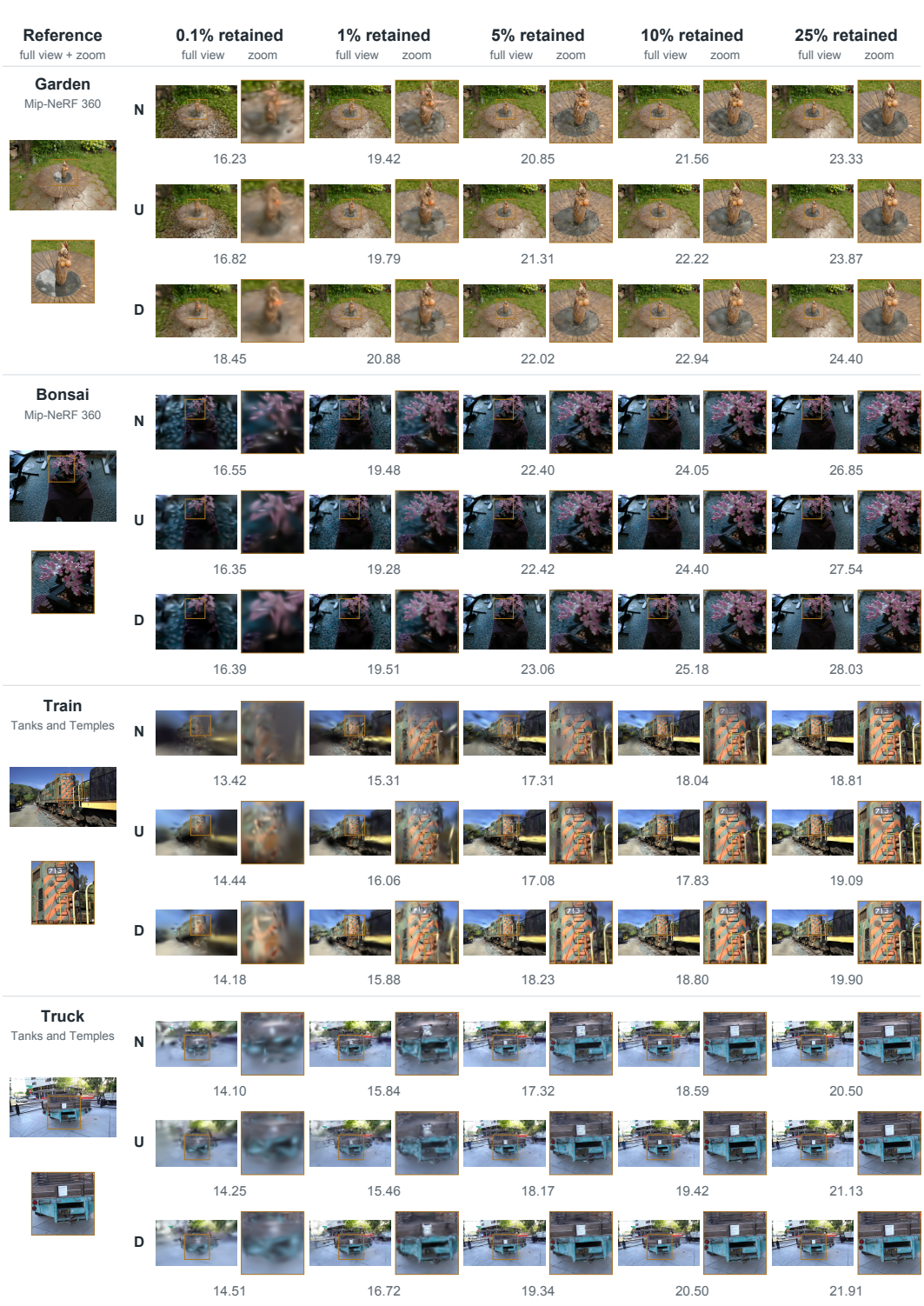

Figure 5: Additional qualitative comparisons at all five retention budgets. Each scene uses one fixed test view and crop; method rows and full-view PSNR are defined in the shared legend. (a) Garden, Bonsai, Train and Truck. Continued on the next page.

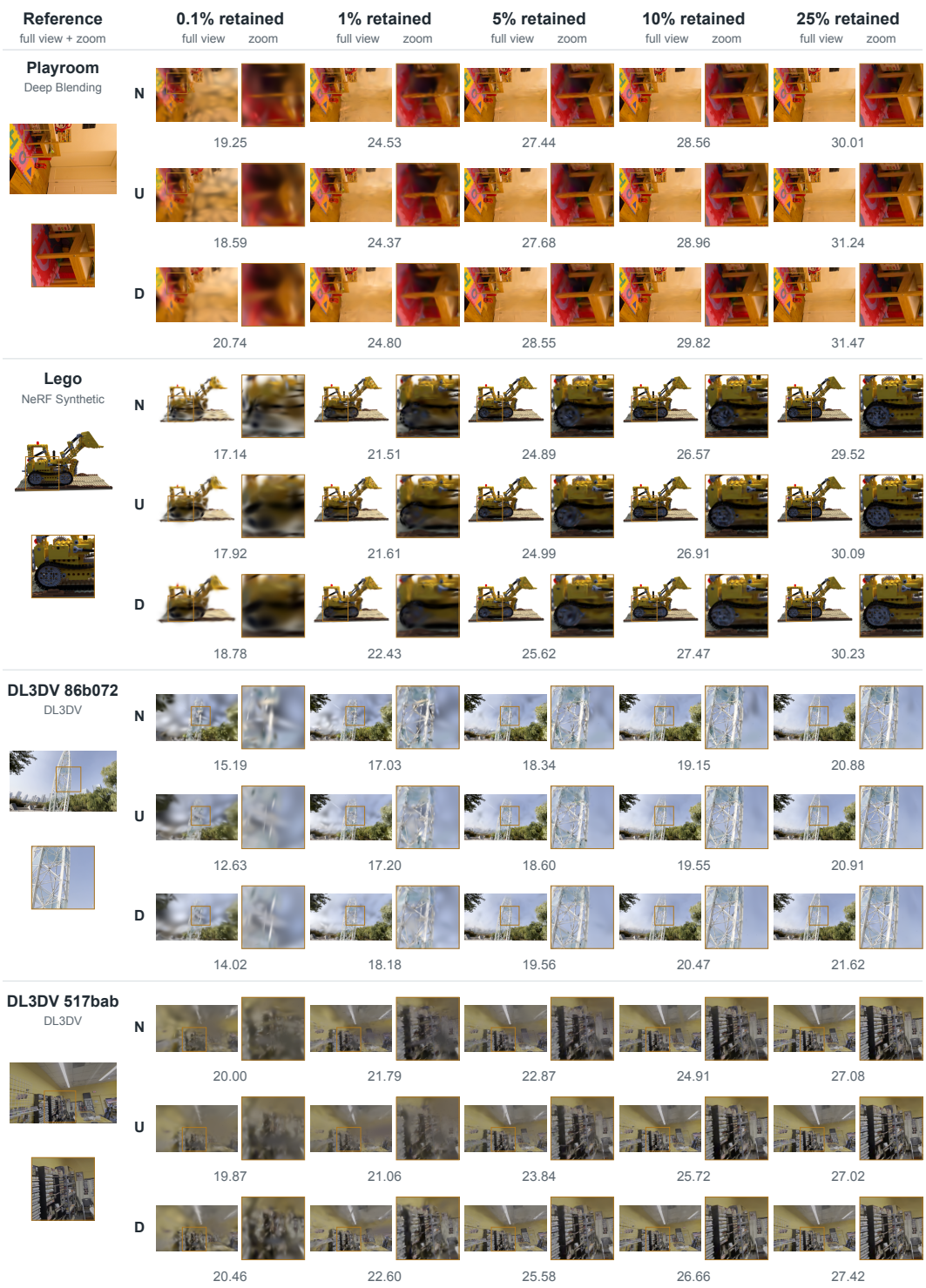

Figure 5 (continued): (b) Playroom, Lego and the two DL3DV scenes. The same method rows, budgets, full-view PSNR and fixed-crop conventions apply.