RATE-ADAPTIVE CONTEXT EVOLUTION FOR PROGRESSIVE 3D GAUSSIAN SPLATTING COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Progressive 3D Gaussian Splatting (3DGS) compression decodes a hierarchy of refinement levels from a single bitstream. However, existing progressive codecs largely rely on entropy contexts whose content or spatial granularity does not adapt to the level-dependent distribution of active Gaussians, limiting probability estimation. We introduce Rate-Adaptive Context Evolution (RACE), which treats the entropy context as a causal state that evolves with decoded content. RACE first splats previously decoded Gaussian attributes onto directional hint maps to update a shared triplane prior. It then selects a directional granularity tuple using encoder-side rate estimates and transmits the selected indices, allowing the decoder to reconstruct the same context without repeating the search. Finally, cross-level attention uses the preceding-level anchor state to condition the aggregation of the current-level spatial neighborhood. Built on the progressive framework of PCGS, RACE produces multiple rates from a single bitstream. Across 19 scenes from four benchmarks, RACE improves the rate–distortion performance of PCGS while keeping its decoding time nearly unchanged, and remains competitive with recent 3DGS codecs. These results demonstrate the benefit of adapting context content, spatial scale, and cross-level interaction for progressive 3DGS entropy modeling.

1 INTRODUCTION

Single-rate compression of 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023), an explicit alternative to NeRF (Mildenhall et al., 2022), is well studied. Compact anchors, pruning, quantization, and learned entropy models achieve strong rate–distortion performance (Lu et al., 2024; Chen et al., 2024; Wang et al., 2024b; Zhan et al., 2025a). However, progressive 3DGS compression, which reconstructs a scene at successively higher quality, remains less explored (Chen et al., 2026). Existing progressive methods mainly focus on constructing scalable layers and supporting multiple rates, but largely inherit context models from single-rate codecs without considering the level-dependent distributional characteristics of Gaussian primitives. The resulting context can become less representative of the current level and reduce probability-estimation accuracy. This work therefore addresses three aspects of progressive context modeling: how to update the prior, how to adapt its granularity to each level, and how to jointly exploit intra- and inter-level context for probability estimation.

In existing progressive codecs, the static prior manifests in two forms. First, some methods share the same spatial prior across all levels, as illustrated in Fig. 1(a), without considering distribution differences introduced by new Gaussians. Second, PCGS conditions each existing anchor on its own previous-level value (Chen et al., 2026), as shown in Fig. 1(b). Consequently, newly activated anchors are encoded without a cross-level prior. Neither design constructs an updated spatial prior that supports all anchors at the next level.

Another problem is that the context granularity is fixed: codecs predefine a set of context scales (Chen et al., 2024) rather than adapting them to the Gaussian distribution of each level. As each level activates new anchors and refines existing ones, the coded symbols differ in density and statistics. A fixed schedule may then merge distinct structures at coarse scales or disperse the available evidence at fine scales, mismatching the granularity to the level's distribution, as shown in Fig. 1(a). By contrast, we select each level's granularity according to estimated rates, as shown in Fig. 1(d).

Furthermore, existing methods exploit intra- and inter-level dependencies largely as separate context sources. ContextGS concatenates a hyperprior (Ballé et al., 2018) with features from decoded coarser

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.

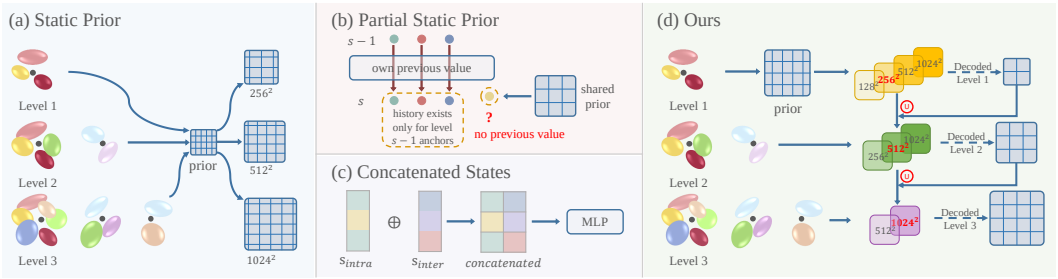


Figure 1: **Motivation.** (a) Existing codecs resample one prior for every level. (b) PCGS Chen et al. (2026) conditions each anchor on its own previous-level value, so newly activated anchors (dashed) fall back on a static prior. (c) Existing entropy models combine intra- and inter-level states only at a late prediction stage, without using one to revise the other. (d) The proposed RACE progressively utilizes the decoded evidence to update the prior and adaptively selects the granularity at each level based on the estimated rate (red marked).

anchors to construct an intra-level spatial context (Wang et al., 2024b), whereas PCGS uses an anchor's own previous-level value as an inter-level prior (Chen et al., 2026). However, the preceding-level state does not explicitly influence how current-level spatial evidence is aggregated, leaving the interaction between spatial structure and progressive refinement history underexplored (Fig. 1(c)). RACE introduces this interaction before distribution estimation. Specifically, the preceding-level anchor feature serves as a query over the current-level spatial neighborhood, allowing inter-level evidence to modulate the aggregation of intra-level context. The resulting conditioned spatial feature, together with the preceding-level state, is then provided to the entropy model for probability estimation. Thus, rather than merely combining two context sources at the prediction head, RACE uses cross-level evidence to adapt the construction of the current-level spatial context itself.

In PCGS, a representative progressive coding framework, 96.1%–98.6% of the active anchors at each level require further attribute refinement, while 1.4%–3.9% are newly activated (Chen et al., 2026). These changes alter the anchor distribution across levels, making a static context mismatched with the current representation. Therefore, we propose Rate-Adaptive Context Evolution (RACE), which adapts the context through three coordinated steps. First, RACE updates the spatial prior with hint maps derived from decoded Gaussians, incorporating newly available cross-level evidence. Second, it selects directional context granularities according to estimated rates, adapting the context scale to distribution in each level. Third, it integrates spatial contexts from the current and preceding levels through hierarchical context attention, jointly capturing intra- and inter-level dependencies for probability estimation. Together, these steps evolve the context in its content, granularity, and dependency modeling as progressive coding proceeds. Our contributions are summarized as follows:

- We formulate progressive 3DGS entropy context as a causal state whose content and spatial granularity should evolve with the decoded Gaussian distribution, exposing a mismatch in existing static-context designs.
- We instantiate this principle with RACE, which updates a shared triplane prior from decoded Gaussians, selects directional granularities according to estimated coding rate, and uses the preceding-level state to condition current-level spatial aggregation while preserving decoder causality.
- Experiments on 19 scenes from four benchmarks show consistent rate–distortion improvements over the underlying progressive codec, together with competitive compression performance against recent 3DGS codecs and decoding cost close to PCGS.

2 Related Work

Progressive 3DGS compression. Progressive 3DGS compression supports multiple reconstruction qualities through a reusable hierarchy. LapisGS uses cumulative layers (Shi et al., 2025), GoDe constructs a gradient-sensitive scalable hierarchy (Di Sario et al., 2025), and PCGS jointly refines anchor masks and attributes across progressive levels (Chen et al., 2026). Related representations also employ progressive propagation or level-of-detail octrees (Cheng et al., 2024; Ren et al., 2025). ProGS

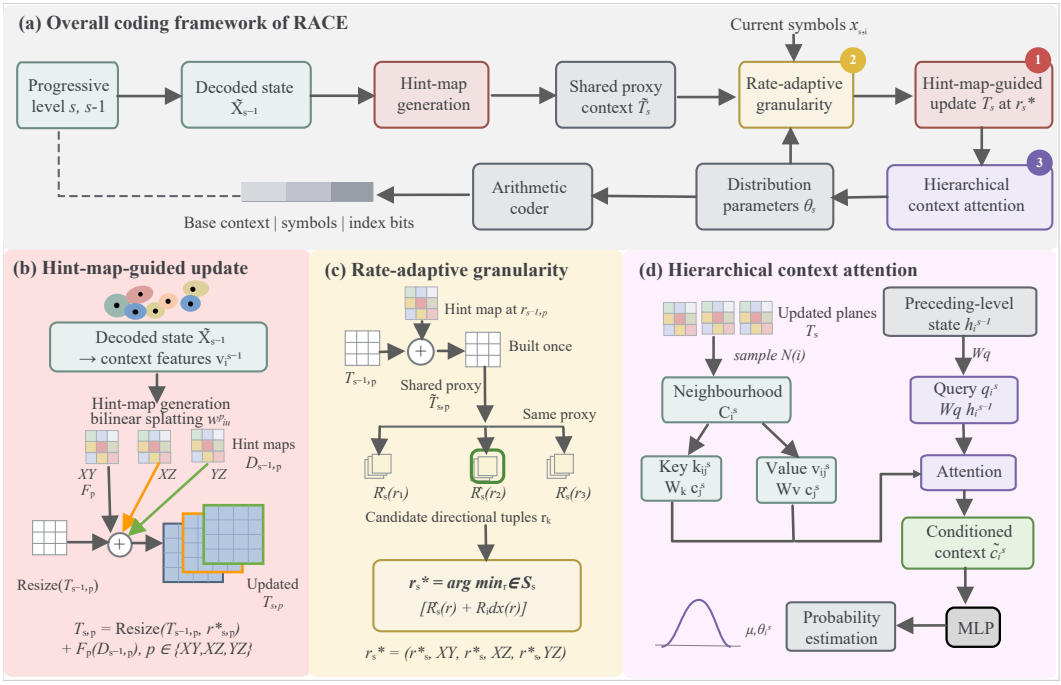


Figure 2: **Overview of RACE.** (a) The overall coding pipeline, where the decoded state from the preceding level is used to evolve the context before probability estimation and arithmetic coding at the current level. (b) Decoded Gaussians update the shared triplane prior through directional hint maps. (c) Directional granularities are selected according to estimated coding rates. (d) The preceding-level state conditions current-level spatial aggregation through hierarchical context attention.

exploits topological parent–child dependencies for entropy coding, using local parent information as cross-level context (Tang et al., 2026). While these methods effectively organize progressive dependencies, the cross-level contexts in PCGS and ProGS are tied primarily to individual anchors or local parent–child relations rather than being used to evolve a shared spatial representation. RACE instead aggregates the complete decoded representation of the preceding level into a shared spatial prior, allowing cross-level evidence to condition both persistent and newly activated anchors at the current level.

Context-based 3DGS compression. Compact 3DGS methods use pruning, quantization, and learned attributes (Lee et al., 2024; Girish et al., 2024; Fan et al., 2024; Liu et al., 2024b; Niedermayr et al., 2024; Wang et al., 2025a), with rate–distortion training and hybrid entropy models further improving coding (Wang et al., 2024a; Liu et al., 2024a). HAC and HAC++ predict anchor attributes from hash-grid context and intra-anchor correlations (Chen et al., 2024; 2025b); ContextGS and FCGS model coarse-to-fine or inter-/intra-Gaussian dependencies (Wang et al., 2024b; Chen et al., 2025a). CAT-3DGS and TC-GS use triplane context (Zhan et al., 2025a; Wang et al., 2025b), related to K-Planes and progressive tri-plane coding (Fridovich-Keil et al., 2023; Jeon et al., 2023), while CAT-3DGS Pro adopts a PCA-guided vector–matrix hyperprior (Zhan et al., 2025b). These single-rate contexts are not updated from all Gaussians decoded at each progressive level.

3 METHOD

RACE is built upon the progressive coding framework of PCGS (Chen et al., 2026) and retains its progressive masking and quantization strategies. As illustrated in Fig. 2(a), RACE evolves the context from the preceding-level decoded state before current-level entropy coding. Decoded Gaussians first update the shared triplane prior through directional hint maps (Fig. 2(b)), after which directional granularities are selected according to estimated coding rates (Fig. 2(c)). Hierarchical context attention then uses the preceding-level state to condition current-level spatial aggregation for probability estimation and arithmetic coding (Fig. 2(d)). The selected granularity indices are

transmitted in the bitstream so that the decoder can reconstruct the same context without repeating the rate search.

3.1 HINT-MAP-GUIDED CONTEXT UPDATE

Before coding level$_s$, both the encoder and decoder have reconstructed the anchors from level$_{s-1}$. Their positions and attributes provide new evidence about the current scene representation, which RACE uses to revise the spatial prior for the next level rather than reusing a fixed one.

We represent the spatial prior by three triplanes because Gaussian primitives exhibit different spatial distributions along the XY, XZ, and YZ directions. The three projections preserve direction-specific spatial relationships and allow each direction to use a different granularity, offering greater flexibility than a single isotropic grid. The base planes $\{\mathbf{T}_{0,p}\}_{p\in\{XY,XZ,YZ\}}$ are reconstructed before progressive entropy decoding.

For anchor i, a transform ϕ maps its decoded attributes $\hat{\mathbf{a}}_i^{s-1}$ to a context feature $\mathbf{v}_i^{s-1}$. We project these features onto plane p and aggregate them at pixel u with bilinear weights to form the hint map:

$$\mathbf{D}_{s-1,p}(u) = \frac{\sum_{i\in\mathcal{I}_{s-1}} w_{iu}^{p}\mathbf{v}_i^{s-1}}{\sum_{i\in\mathcal{I}_{s-1}} w_{iu}^{p} + \epsilon}, \tag{1}$$

where w_{iu}^{p} denotes the contribution of anchor i to pixel u, and ϵ prevents division by zero. This operation uses only information decoded before level$_s$.

After Sec. 3.2 selects the directional granularity $r_{s,p}^{\star}$, the decoded features are projected at that granularity; we write $\mathbf{D}_{s-1,p}^{(r)}$ for the hint map projected at granularity r. The context used to code level$_s$ is:

$$\mathbf{T}_{s,p} = \mathrm{Resize}(\mathbf{T}_{s-1,p}, r_{s,p}^{\star}) + F_p\left(\mathbf{D}_{s-1,p}^{(r_{s,p}^{\star})}\right). \tag{2}$$

The first term carries the spatial prior used at the preceding level to the selected granularity, while the plane-specific transform F_p injects the newly decoded evidence. We denote the resulting triplane context by $\mathbf{T}_s = \{\mathbf{T}_{s,p}\}_p$. This recursive update changes both the content and the directional granularity of the context as the progressive representation is refined.

3.2 RATE-ADAPTIVE DIRECTIONAL GRANULARITY

A finer granularity is not necessarily more effective. A coarse plane may merge distinct spatial structures, whereas an excessively fine plane can disperse the projected evidence across sparsely populated locations. We therefore select the directional granularity according to its estimated coding rate rather than a predefined level-to-granularity schedule.

Rate estimation and actual coding use different context-generation paths. For rate estimation, the decoded features from level$_{s-1}$ are projected once at the granularity already used by the inherited context. This gives a shared proxy context:

$$\widetilde{\mathbf{T}}_{s,p} = \mathbf{T}_{s-1,p} + F_p\left(\mathbf{D}_{s-1,p}^{(r_{s-1,p})}\right), \tag{3}$$

which is constructed only once per level. Let $\mathcal{Q} = \{128, 256, 512, 1024\}$ be the resolution alphabet and $\mathbf{r} = (r_{XY}, r_{XZ}, r_{YZ})$ a directional granularity tuple. At level s, the feasible set is $\mathcal{S}_s = \mathcal{Q}_{s,XY} \times \mathcal{Q}_{s,XZ} \times \mathcal{Q}_{s,YZ}$, where each $\mathcal{Q}_{s,p} \subseteq \mathcal{Q}$ retains resolutions between the preceding-level resolution and the resolution used for training at that level. Candidate contexts $\widetilde{\mathbf{T}}_s(\mathbf{r})$ are derived from the same proxy context by resizing each plane to the candidate granularity, without regenerating the hint maps. Let $y_{s,i}$ denote the quantized attributes of anchor i at level$_s$, and let $\hat{\mathbf{X}}_{s-1}$ denote the representation reconstructed through level $s-1$.

For each $\mathbf{r} \in \mathcal{S}_s$, the encoder estimates:

$$\hat{R}_s(\mathbf{r}) = -\sum_i \log_2 p_\theta\left(y_{s,i} \mid \widetilde{\mathbf{T}}_s(\mathbf{r}), \hat{\mathbf{X}}_{s-1}, \lambda_s\right). \tag{4}$$

$\hat{R}_s(\mathbf{r})$ is the rate predicted by the entropy model of Sec. 3.3, evaluated on $\widetilde{\mathbf{T}}_s(\mathbf{r})$ with query features $\mathbf{h}_i^{s-1}$ drawn from $\hat{\mathbf{X}}_{s-1}$; λ_s identifies the current progressive level. The encoder selects:

$$\mathbf{r}_s^\star = \arg\min_{\mathbf{r}\in\mathcal{S}_s} \left[\hat{R}_s(\mathbf{r}) + R_{\text{idx}}(\mathbf{r})\right], \tag{5}$$

where R_{idx} accounts for signaling the three selected indices. With $|\mathcal{Q}| = 4$, this costs six fixed-length bits per refinement level. The level embedding defines the coding task, but the current attributes and proxy context determine which directional tuple minimizes its rate.

The encoder evaluates every tuple in $\mathcal{S}_s$ through entropy-model forward passes. For actual coding, the decoded Gaussians are projected at the selected granularities, and Eq. (2) constructs the final context. Arithmetic coding is executed only for this context; the other candidates require only entropy-model forward passes. The encoder writes the selected indices to the bitstream, and the decoder directly reconstructs the same context from them. The proxy-based estimate remains well calibrated with the actual arithmetic-coded payload (Appendix E).

3.3 Hierarchical Context Attention

After selecting $\mathbf{r}_s^\star$, we sample the evolved triplanes at the anchors in the neighborhood $\mathcal{N}(i)$ of anchor i. Their features $\mathbf{C}_i^s = \{\mathbf{c}_j^s\}_{j\in\mathcal{N}(i)}$ form the intra-level context and describe the current spatial structure. In contrast, the preceding-level feature $\mathbf{h}_i^{s-1}$ records how the anchor has been represented before the current refinement and serves as the inter-level context. We use the preceding-level state to condition the aggregation of the current-level neighborhood. Because newly activated anchors have no corresponding state at level $s-1$, we first define a validity indicator $m_i^{s-1} \in \{0,1\}$, where $m_i^{s-1} = 1$ if anchor i has a valid preceding-level state and $m_i^{s-1} = 0$ otherwise. We then define the effective preceding-level state as:

$$\bar{\mathbf{h}}_i^{s-1} = m_i^{s-1}\mathbf{h}_i^{s-1} + (1 - m_i^{s-1})\mathbf{h}_{\text{null}}, \tag{6}$$

where $\mathbf{h}_{\text{null}}$ is a zero vector for anchors without a preceding-level state. The effective preceding-level state produces the query, whereas the current-level neighboring features produce the keys and values:

$$\begin{aligned} \widetilde{\mathbf{c}}_i^s &= \sum_{j\in\mathcal{N}(i)} \text{softmax}_j\left(\frac{(\mathbf{q}_i^s)^\top \mathbf{k}_j^s}{\sqrt{d}}\right)\mathbf{v}_j^s, \\ \mathbf{q}_i^s &= W_q\bar{\mathbf{h}}_i^{s-1}, \\ \mathbf{k}_j^s &= W_k\mathbf{c}_j^s, \qquad \mathbf{v}_j^s = W_v\mathbf{c}_j^s. \end{aligned} \tag{7}$$

where d is the feature dimension. Thus, for an anchor with a valid refinement history, $\widetilde{\mathbf{c}}_i^s$ is not an independently aggregated spatial feature: its attention weights are explicitly conditioned on the preceding-level state. For a newly activated anchor, the null state removes anchor-specific cross-level information, and the attention is determined solely from its current-level spatial neighborhood. The entropy model uses the conditioned spatial context together with the effective preceding-level state and the level embedding to predict the distribution parameters:

$$\boldsymbol{\theta}_i^s = F_{\text{ent}}\left(\widetilde{\mathbf{c}}_i^s, \bar{\mathbf{h}}_i^{s-1}, \lambda_s\right). \tag{8}$$

These parameters define the distribution $p_{\theta_i^s}$ used for entropy coding and for the rate estimation in Eq. (4).

Although newly activated anchors lack an anchor-specific refinement history, they still access the shared spatial prior updated from all Gaussians decoded at level $s-1$. The two cross-level paths therefore serve complementary roles: the decoded-Gaussian update provides a shared spatial prior to every current-level anchor, whereas hierarchical attention uses anchor-specific refinement history only when such history is available.

4 Experiments

4.1 Experimental Setup

Datasets and metrics. We evaluate 19 scenes from Mip-NeRF 360, Tanks & Temples, Deep Blending, and BungeeNeRF (Barron et al., 2022; Knapitsch et al., 2017; Hedman et al., 2018;

Xiangli et al., 2022). We report PSNR, SSIM (Wang et al., 2004), LPIPS (Zhang et al., 2018), and compressed size in decimal MB. Finally, these individual metrics are combined into the compression rank defined by 3DGS.zip (Bagdasarian et al., 2025), which jointly summarizes reconstruction quality and compressed size; lower is better. Complete metrics and operating points appear in Appendix B.

Baselines. We compare with HAC (Chen et al., 2024), HAC++ (Chen et al., 2025b), ContextGS (Wang et al., 2024b), CAT-3DGS (Zhan et al., 2025a), and the progressive baselines PCGS (Chen et al., 2026) and ProGS (Tang et al., 2026). GoDe (Di Sario et al., 2025) and HEMGS (Liu et al., 2024a) are included where published results overlap our datasets; missing coverage is marked.

Implementation. We retain the dataset-specific progressive schedules and optimization settings of PCGS. Mip-NeRF 360, Tanks & Temples, and Deep Blending are trained for 30k iterations, whereas BungeeNeRF uses 40k iterations in the main comparison. Each stream exposes three or four operating points, with directional granularities selected from $\mathcal{Q} = \{128, 256, 512, 1024\}$. Ablations share the same initialization, training schedule, entropy model, and evaluation code. Appendix A gives the full configuration and objective.

4.2 Rate-Distortion Performance

Fig. 3 compares progressive PSNR–size curves on Mip-NeRF 360. From one stream, RACE remains competitive with single-rate codecs and consistently shifts the PCGS curve toward lower rates. Over their common PSNR ranges, RACE reduces the rate relative to PCGS by 27.37%, 13.21%, and 16.69% BD-rate (Bjøntegaard, 2001) on Mip-NeRF 360, Deep Blending, and BungeeNeRF, respectively. On Tanks and Temples, the measured RACE curve lies above the maximum PCGS PSNR, so no common interval exists and BD-rate is not reported. The overlap ranges and operating points are given in Appendix B. Table 1 provides a complementary aggregate summary by selecting one measured operating point for each method and dataset and applying the compression-rank protocol described in Appendix B.1. Among methods with complete coverage of all four datasets, RACE obtains the lowest aggregate rank. Because this rank depends on both dataset coverage and the selected operating point, we use it as a summary statistic rather than as the primary rate–distortion evidence.

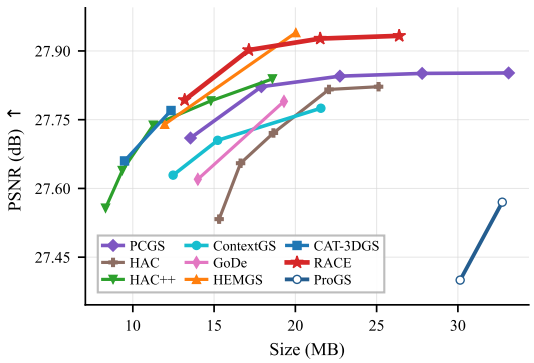


Figure 3: RD curves on Mip-NeRF 360.

Qualitative comparison. Fig. 4 compares similar-size reconstructions on four scenes, with crops showing thin structures and isolated details. Additional examples appear in Appendix C. In Flowers, RACE better preserves the narrow foreground boundary; in Room, it retains the sharp transition at the leaf edge. These localized differences are consistent with the quantitative gains and are not caused by a larger stream.

4.3 Ablation Analysis

Fig. 5 and Table 2 evaluate the three components over their complete measured curves. The figure organizes variants by component, while the table uses the complete RACE configuration as a common BD-rate reference for all rows.

Decoded-Information-Guided Context. Fig. 5 (left) isolates decoded-context updates under the same granularity schedule and previous–current context. The update improves the 19-scene curve throughout the measured range. Relative to RACE, the variants without and with decoded context require 45.01% and 26.00% more rate, respectively, when averaged over their available dataset overlaps (Table 2). This consistent shift indicates that decoded Gaussians provide spatial evidence that is absent from a context inherited unchanged from the preceding level.

Table 1: Rank aggregates the available datasets following Eq. (12). Red and blue bold denote the best and second-best displayed values; the best Rank is also highlighted in yellow.

Dataset	Metric	PCGS	HAC	HAC++	ContextGS	GoDe	HEMGS	CAT-3DGS	ProGS	RACE
Mip-NeRF 360	PSNR↑	27.852	27.822	27.839	27.775	27.79	**27.94**	27.77	27.29	**27.902**
	SSIM↑	0.811	**0.812**	**0.812**	0.811	0.81	**0.813**	0.809	**0.813**	0.81
	LPIPS↓	0.233	**0.229**	0.23	0.23	0.284	0.23	0.241	**0.219**	0.238
	Size↓	33.12	25.12	18.59	21.57	19.3	20.03	**12.35**	20.84	**17.14**
Tanks & Temples	PSNR↑	24.295	24.318	24.325	24.149	24.01	**24.55**	**24.41**	24.24	24.299
	SSIM↑	0.854	**0.856**	**0.855**	0.854	0.848	**0.856**	0.853	**0.855**	0.852
	LPIPS↓	0.179	**0.174**	0.176	0.18	0.224	0.176	0.189	**0.167**	0.185
	Size↓	17.89	13.04	8.54	8.33	11	9.67	**6.93**	10.56	**6.35**
Deep Blending	PSNR↑	30.303	30.217	30.263	30.253	**30.33**	30.24	30.29	–	**30.333**
	SSIM↑	**0.91**	**0.909**	**0.909**	**0.909**	**0.909**	**0.909**	**0.909**	–	**0.91**
	LPIPS↓	**0.257**	**0.255**	0.26	0.26	0.331	0.27	0.269	–	0.259
	Size↓	7.25	6.61	3.99	4.38	8.4	**2.85**	**3.56**	–	6.51
BungeeNeRF	PSNR↑	26.838	26.483	**27.065**	**27.051**	–	–	26.65	25.43	26.993
	SSIM↑	0.865	0.847	**0.872**	0.87	–	–	0.859	0.855	**0.873**
	LPIPS↓	0.22	0.248	0.211	0.217	–	–	0.226	**0.164**	**0.207**
	Size↓	**14**	18.58	16.17	15.39	–	–	16.21	18.14	**15.24**
	Rank↓	5.333	5.813	**3.521**	4.667	6.75	3.611	4.313	5.056	**3.292**

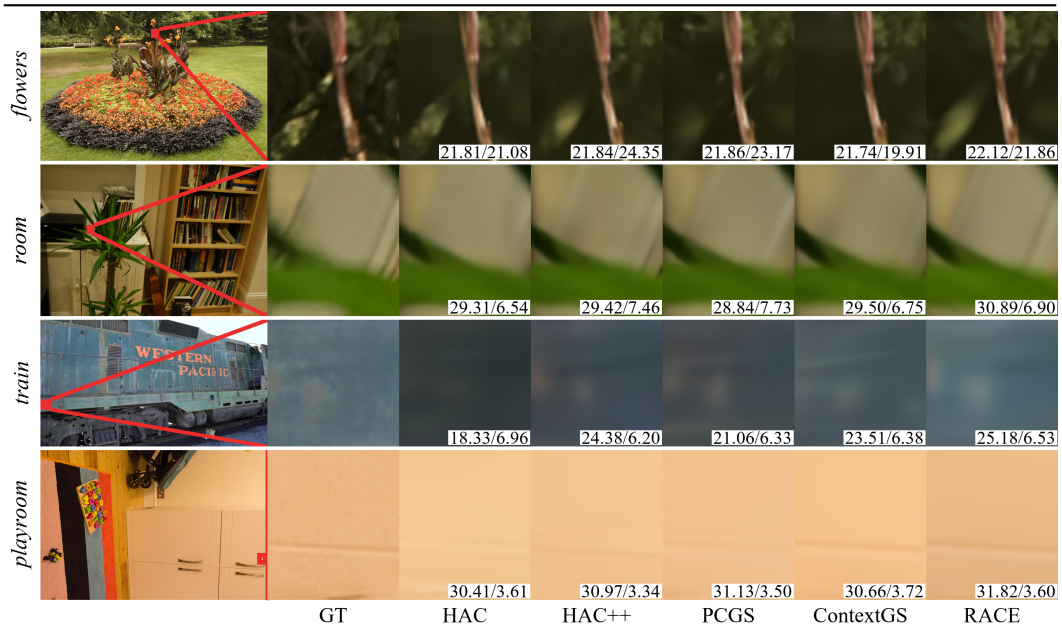


Figure 4: Qualitative comparison on Flowers, Room, Train, and Playroom. The labels report

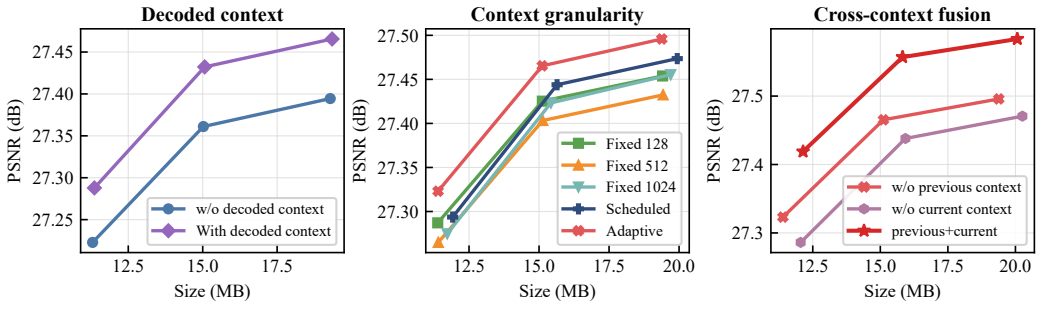
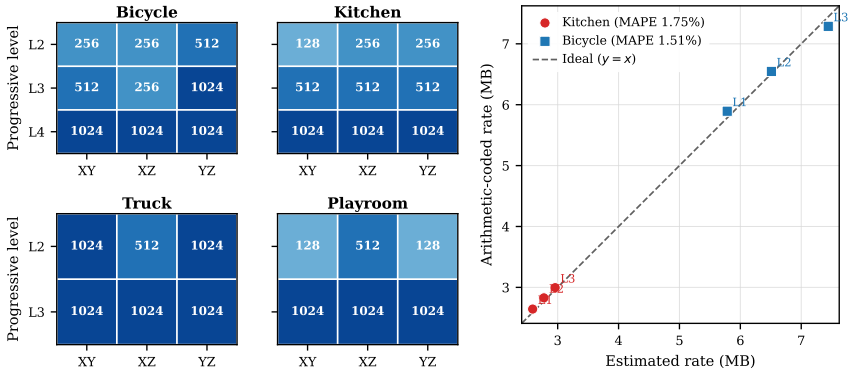


Figure 5: PSNR–size ablations averaged over all 19 scenes: hint-map-guided update (left), directional granularity (middle), and hierarchical context attention (right).

Table 2: PSNR BD-rate (%) of each ablation relative to RACE. Positive values are extra rate at equal quality; "–" indicates no common PSNR interval. C and P denote current- and previous-level context.

Variant	Dec.	Gran.	Context	Mip	T&T	DB	Bungee	Avg.
Hint-map-guided context update								
w/o decoded context	×	Scheduled	P+C	–	–	–	45.01	45.01
With decoded context	✓	Scheduled	P+C	49.68	–	2.46	25.87	26.00
Rate-adaptive directional granularity								
Fixed 128	✓	128	C	42.54	–	4.32	33.60	26.82
Fixed 512	✓	512	C	53.19	–	21.18	49.28	41.21
Fixed 1024	✓	1024	C	36.36	77.25	31.75	37.96	45.83
Scheduled	✓	Scheduled	C	45.33	41.92	17.05	25.38	32.42
Adaptive	✓	Adaptive	C	26.05	38.48	10.28	17.51	23.08
Hierarchical context attention								
w/o previous context	✓	Adaptive	C	26.05	38.48	10.28	17.51	23.08
w/o current context	✓	Adaptive	P	39.50	–	11.12	38.64	29.76
RACE	✓	Adaptive	P+C	0.00	0.00	0.00	0.00	0.00



(a) Selected directional granularities **(b) Rate-estimation accuracy**

Figure 6: Granularity diagnostics. (a) Selected directional granularities. (b) Estimated versus arithmetic-coded rates.

Rate-Adaptive Directional Granularity. Fig. 5 (middle) compares fixed, scheduled, and adaptive granularities with current-only context. No fixed choice dominates the full rate range. Relative to RACE, the fixed and scheduled variants require 26.82%–45.83% more rate on average, while adaptive granularity with current-only context requires 23.08% more rate.

(1) Selected granularities. Fig. 6(a) shows that choices vary by level, scene, and direction. All scenes select 1024 at the final level, but Playroom and Truck choose $(128, 512, 128)$ and $(1024, 512, 1024)$ at level$_2$; Bicycle uses $(512, 256, 1024)$ at level$_3$. Coarse choices can minimize rate because overly fine planes disperse decoded evidence across more cells. These patterns support directional, level-wise selection rather than a fixed schedule.

(2) Rate-proxy accuracy. We compare the entropy estimate with the arithmetic-coded feature and scaling payload at three refinement levels of Kitchen and Bicycle. Accuracy is measured by mean absolute percentage error (MAPE),

$$\mathrm{MAPE} = \frac{100\%}{N} \sum_{i=1}^{N} \left| \frac{\widehat{R}_i - R_i}{R_i} \right|, \tag{9}$$

where $\widehat{R}_i$ and R_i denote estimated and coded rates. Fig. 6(b) reports 1.75% and 1.51% MAPE for Kitchen and Bicycle, respectively. The small residual partly reflects file headers included only in the coded payload. Because selection compares candidates within the same level, this agreement is sufficient for reliable ordering even when a small absolute offset remains.

Cross-Level Context Modeling. Fig. 5 (right) shows that current- and previous-level evidence are complementary. Removing either source costs 23.08% and 29.76% BD-rate on average over valid comparisons (Table 2). Current-level neighborhoods describe local spatial structure, whereas the previous-level feature identifies how an anchor refines its decoded state. The loss from either ablation supports conditioning on both sources.

Additional diagnostics are reported in Appendices F.3 and F.2. At matched quality on Quebec, RACE reduces the total size by 24.1% relative to CAT-3DGS, while refined anchors account for 93.9% of its refinement payload. The component decomposition is treated as diagnostic evidence rather than a controlled rate–distortion comparison.

4.4 Runtime and Coding Cost

Table 3 reports wall-clock cost on one RTX 5880 Ada GPU. HAC, HAC++, and ContextGS produce one rate per run; PCGS and RACE produce a multi-rate stream, so coding times should be compared within these groups. RACE averages 28.1 s encoding, 27.8 s decoding, and 153.1 FPS, versus 20.4 s, 27.4 s, and 178.6 FPS for PCGS. It remains faster than ContextGS (96.0 s, 85.5 s, and 84.5 FPS). Baseline points follow Table 1 and all averages cover the same 19 scenes. Thus, RACE adds 37.7% encoding time but only 1.5% decoding time over PCGS. The selected granularities are read directly from the bitstream, so the encoder-side search is not repeated during decoding.

Table 3: Runtime on an RTX 5880 Ada GPU. Enc. and Dec. are measured in seconds. For FPS, yellow and pink indicate the best and second-best results.

	Mip-NeRF 360			Tanks & Temples			Deep Blending			BungeeNeRF		
Method	Enc.	Dec.	FPS	Enc.	Dec.	FPS	Enc.	Dec.	FPS	Enc.	Dec.	FPS
HAC	7.6	13.5	119.6	3.7	6.6	142.1	2.7	4.5	188.8	7.5	12.9	**191.9**
HAC++	16.4	28.2	119.8	7.7	12.6	163.5	5.4	8.4	189.7	16.5	26.1	177.3
ContextGS	104.7	97.4	67.2	86.8	65.4	98.9	52.6	42.2	84.9	100.5	88.6	105.5
PCGS	22.5	31.3	**135.0**	12.4	16.5	**164.5**	7.2	9.2	**220.4**	24.4	31.3	**234.7**
RACE	32.5	32.6	**125.8**	23.4	17.8	**171.4**	9.2	10.2	**207.4**	29.2	31.6	169.9

Why rate-driven selection is affordable. Rate estimation reuses one context state constructed from the preceding level and one hint map; only the selected tuple triggers the actual update and arithmetic coding. Each level evaluates all feasible tuples in $\mathcal{S}_s$ through entropy-model forward passes and performs one final coding pass, rather than reconstructing the hint maps for every candidate. The decoder reads the selected tuple and reconstructs the context once, so the search adds no decoder-side search.

5 Conclusion

We presented RACE, a progressive 3DGS compression framework that treats entropy context as a causal state evolving with decoded scene content. RACE updates a shared spatial prior from previously decoded Gaussians, adapts directional context granularity using encoder-side rate estimates, and uses cross-level evidence to condition current spatial aggregation. Experiments across four benchmarks show consistent rate–distortion improvements over the underlying progressive codec and competitive compression against recent 3DGS methods, while keeping decoder-side overhead small. A current limitation is the additional encoder cost required for candidate-rate evaluation and the use of a discrete granularity candidate set. Learning the context resolution jointly with the entropy model is an interesting direction for future work.

AI use statement

Generative AI tools were used for research brainstorming, manuscript structuring, and language assistance. The authors reviewed all AI-assisted text and technical suggestions, verified the relevant

literature, and independently validated the implementation and experimental results. The authors take responsibility for the final manuscript.

REFERENCES

Milena T. Bagdasarian, Paul Knoll, Yi-Hsin Li, Florian Barthel, Anna Hilsmann, Peter Eisert, and Wieland Morgenstern. 3DGS.zip: A survey on 3D gaussian splatting compression methods. *Computer Graphics Forum*, 44(2):e70078, 2025. doi: 10.1111/cgf.70078.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations*, 2018.

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*, pp. 5470–5479, 2022.

Gisle Bjøntegaard. Calculation of average PSNR differences between RD curves. In *ITU-T Video Coding Experts Group Meeting*, 2001. Document VCEG-M33.

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*, pp. 422–438, 2024. doi: 10.1007/978-3-031-72661-3_24.

Yihang Chen, Qianyi Wu, Mengyao Li, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Fast feedforward 3D gaussian splatting compression. In *International Conference on Learning Representations*, 2025a.

Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. HAC++: Towards 100x compression of 3D gaussian splatting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(11):10210–10226, 2025b. doi: 10.1109/TPAMI.2025.3594066.

Yihang Chen, Mengyao Li, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. PCGS: Progressive compression of 3D gaussian splatting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 3111–3119, 2026.

Kai Cheng, Xiaoxiao Long, Kaizhi Yang, Yao Yao, Wei Yin, Yuexin Ma, Wenping Wang, and Xuejin Chen. GaussianPro: 3D gaussian splatting with progressive propagation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 8123–8140, 2024.

Francesco Di Sario, Riccardo Renzulli, Marco Grangetto, Akihiro Sugimoto, and Enzo Tartaglione. GoDe: Gaussians on demand for progressive level of detail and scalable compression. *arXiv preprint arXiv:2501.13558*, 2025.

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*, volume 37, pp. 140138–140158, 2024. doi: 10.52202/079017-4447.

Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-Planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12479–12488, 2023.

Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. EAGLES: Efficient accelerated 3D gaussians with lightweight encodings. In *European Conference on Computer Vision*, pp. 54–71, 2024. doi: 10.1007/978-3-031-73036-8_4.

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): 257:1–257:15, 2018. doi: 10.1145/3272127.3275084.

Seungmin Jeon, Kwang Pyo Choi, Youngo Park, and Chang-Su Kim. Context-based trit-plane coding for progressive image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14348–14357, 2023. doi: 10.1109/CVPR52729.2023.01379.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):139:1–139:14, 2023. doi: 10.1145/3592433.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4):78:1–78:13, 2017. doi: 10.1145/3072959.3073599.

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

Lei Liu, Zhenghao Chen, and Dong Xu. HEMGS: A hybrid entropy model for 3D gaussian splatting data compression. *arXiv preprint arXiv:2411.18473*, 2024a.

Xiangrui Liu, Xinju Wu, Pingping Zhang, Shiqi Wang, Zhu Li, and Sam Kwong. CompGS: Efficient 3D scene representation via compressed gaussian splatting. In *Proceedings of the ACM International Conference on Multimedia*, pp. 2936–2944, 2024b. doi: 10.1145/3664647.3681131.

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*, pp. 20654–20664, 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. *Communications of the ACM*, 65(1):99–106, 2022. doi: 10.1145/3503250.

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*, pp. 10349–10358, 2024.

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*, pp. 1–15, 2025. doi: 10.1109/TPAMI.2025.3568201.

Yuang Shi, Géraldine Morin, Simone Gasparini, and Wei Tsang Ooi. LapisGS: Layered progressive 3D gaussian splatting for adaptive streaming. In *International Conference on 3D Vision*, pp. 991–1000, 2025. doi: 10.1109/3DV66043.2025.00096.

Zhiye Tang, Lingzhuo Liu, Shengjie Jiao, Qiudan Zhang, Junhui Hou, You Yang, and Xu Wang. ProGS: Towards progressive coding for 3D gaussian splatting. *arXiv preprint arXiv:2603.09703*, 2026.

Haishan Wang, Mohammad Hassan Vali, and Arno Solin. Smol-GS: Compact representations for abstract 3D gaussian splatting. *arXiv preprint arXiv:2512.00850*, 2025a.

Henan Wang, Hanxin Zhu, Tianyu He, Runsen Feng, Jiajun Deng, Jiang Bian, and Zhibo Chen. End-to-end rate-distortion-optimized 3D gaussian representation. In *European Conference on Computer Vision*, pp. 76–92, 2024a. doi: 10.1007/978-3-031-73636-0_5.

Taorui Wang, Zitong Yu, and Yong Xu. TC-GS: Tri-plane based compression for 3D gaussian splatting. In *IEEE International Conference on Multimedia and Expo*, pp. 1–6, 2025b.

Yufei Wang, Zhihao Li, Lanqing Guo, Wenhan Yang, Alex C. Kot, and Bihan Wen. ContextGS: Compact 3D gaussian splatting with anchor level context model. In *Advances in Neural Information Processing Systems*, volume 37, pp. 51532–51551, 2024b. doi: 10.52202/079017-1631.

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

Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. BungeeNeRF: Progressive neural radiance field for extreme multi-scale scene rendering. In *European Conference on Computer Vision*, pp. 106–122, 2022. doi: 10.1007/978-3-031-19824-3_7.

Yu-Ting Zhan, Cheng-Yuan Ho, Hebi Yang, Yi-Hsin Chen, Jui Chiu Chiang, Yu-Lun Liu, and Wen-Hsiao Peng. CAT-3DGS: A context-adaptive triplane approach to rate-distortion-optimized 3DGS compression. In *International Conference on Learning Representations*, 2025a.

Yu-Ting Zhan, He-bi Yang, Cheng-Yuan Ho, Jui-Chiu Chiang, and Wen-Hsiao Peng. CAT-3DGS Pro: A new benchmark for efficient 3DGS compression. In *European Signal Processing Conference*, pp. 1367–1371, 2025b. doi: 10.23919/EUSIPCO63237.2025.11226320.

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

A Implementation Details

The implementation follows the PCGS (Chen et al., 2026) anchor representation, quantization, progressive ordering, rendering loss, and optimizer configuration. The added path projects already decoded anchor attributes onto the XY, XZ, and YZ planes by bilinear hinting and transforms each projected feature with a plane-specific block. During rate estimation, this projection is performed once at the inherited context granularity and added to that context; all candidate rates are evaluated from the resulting context state. During actual coding, the decoded Gaussians are projected at the selected directional granularity and added to the context used by the preceding level. The entropy head receives the sampled evolved-plane feature, the valid preceding-level anchor state, and the level embedding. Candidate granularities are $\{128, 256, 512, 1024\}$. For each plane, monotonic refinement retains candidates between the preceding-level and training resolutions; their Cartesian product forms $\mathcal{S}_s$. All feasible tuples are evaluated by entropy-model forward passes, while arithmetic coding is invoked only for the selected tuple. Three fixed-length granularity indices (six bits in total) are stored per refinement level. The attention uses four spatial neighbors, one dot-product head of width 64, and a zero preceding-level state for newly activated anchors. Its prediction head contains 76,310 parameters (0.15 MB in FP16), consistent with the 0.14–0.16 MB network allocation measured in Appendix F.3. The measured bitstream additionally stores 0.777 MB for the base triplane and 0.493 MB for all entropy networks; the granularity indices are below 3 bytes per scene. The base context and shared networks are stored once, while higher-level contexts are regenerated causally from decoded Gaussians. Training uses

$$\mathcal{L}_{\mathrm{RD}} = \sum_s \omega_s(\hat{R}_s + \lambda_s D_s), \tag{10}$$

and

$$\mathcal{L}_{\mathrm{ctx}} = \sum_p \|\mathbf{T}_{0,p} - \tilde{\mathbf{T}}_{0,p}\|_1, \quad \mathcal{L} = \mathcal{L}_{\mathrm{RD}} + \gamma \mathcal{L}_{\mathrm{ctx}}, \tag{11}$$

with $\gamma = 0.1$.

B Operating Points for the Rate–Distortion Comparison

Tables 4 and 5 report the complete dataset-average operating points used to organize the rate–distortion results associated with Section 4.2. The RACE rows use the complete decoded-context, adaptive-granularity, previous–current configuration. PSNR BD-rate is computed by monotone interpolation of log-rate only over the quality interval shared with PCGS. The intervals are $[27.793, 27.852]$ dB for Mip-NeRF 360, $[30.135, 30.308]$ dB for Deep Blending, and $[26.993, 27.153]$ dB for BungeeNeRF. Tanks and Temples has no overlap because the lowest measured RACE PSNR exceeds the highest measured PCGS PSNR; we therefore report no extrapolated value.

B.1 Compression Rank

Following the 3DGS.zip compression-rank definition (Bagdasarian et al., 2025), PSNR and SSIM are ranked in descending order, whereas LPIPS and model size are ranked in ascending order. Ties

Table 4: Dataset-average operating points used for the rate–distortion comparison in Section 4.2 on (a) Mip-NeRF 360 and (b) Tanks and Temples. PCGS uses the 30k-iteration results, and the RACE rows use the complete configuration.

	(a) Mip-NeRF 360					(b) Tanks and Temples			
Method	PSNR↑	SSIM↑	LPIPS↓	Size (MB)↓	Method	PSNR↑	SSIM↑	LPIPS↓	Size (MB)↓
PCGS (30k)	27.710	0.8078	0.2370	13.55	PCGS (30k)	24.172	0.8492	0.1848	7.02
	27.822	0.8101	0.2346	17.91		24.274	0.8533	0.1806	9.29
	27.845	0.8106	0.2339	22.74		24.289	0.8540	0.1799	11.97
	27.851	0.8108	0.2335	27.80		24.292	0.8542	0.1795	14.85
	27.852	0.8109	0.2332	33.12		24.295	0.8543	0.1794	17.89
HAC	27.533	0.807	0.239	15.31	HAC	24.171	0.850	0.185	8.05
	27.655	0.809	0.235	16.63		24.270	0.852	0.181	8.76
	27.721	0.810	0.233	18.65		24.275	0.853	0.179	9.98
	27.816	0.811	0.230	22.07		24.318	0.856	0.174	13.04
	27.822	0.812	0.229	25.12					
HAC++	27.557	0.803	0.253	8.32	HAC++	24.213	0.849	0.190	5.37
	27.639	0.805	0.249	9.35		24.304	0.851	0.185	5.99
	27.738	0.807	0.242	11.26		24.249	0.852	0.182	6.97
	27.791	0.810	0.235	14.80		24.325	0.855	0.176	8.54
	27.839	0.812	0.230	18.59		24.401	0.857	0.173	10.27
ContextGS	27.629	0.808	0.237	12.48	ContextGS	24.149	0.854	0.180	8.33
	27.705	0.809	0.233	15.22		24.212	0.854	0.176	10.03
	27.775	0.811	0.230	21.57		24.271	0.856	0.175	11.90
GoDe	27.620	0.803	0.297	14.00	GoDe	24.010	0.848	0.224	11.00
	27.790	0.810	0.284	19.30					
HEMGS	27.740	0.807	0.249	11.96	HEMGS	24.400	0.848	0.192	5.75
	27.940	0.813	0.230	20.03		24.550	0.856	0.176	9.67
CAT-3DGS	27.660	0.807	0.249	9.49	CAT-3DGS	24.220	0.850	0.194	5.47
	27.770	0.809	0.241	12.35		24.410	0.853	0.189	6.93
RACE	27.793	0.8076	0.2406	13.19	RACE	24.299	0.8516	0.1851	6.35
	27.902	0.8099	0.2379	17.14		24.381	0.8549	0.1814	8.24
	27.927	0.8106	0.2368	21.53		24.391	0.8555	0.1805	10.68
	27.933	0.8108	0.2362	26.39					

Rows within each method are ordered by increasing bitstream size. The new ablation averages 9 Mip-NeRF 360 scenes and 2 Tanks and Temples scenes. Sizes are decimal MB.

Table 5: Dataset-average operating points used for the rate–distortion comparison in Section 4.2 on (c) Deep Blending and (d) BungeeNeRF. PCGS uses the 30k-iteration results in (c) and the 40k-iteration results in (d), and the RACE rows use the complete configuration.

	(c) Deep Blending					(d) BungeeNeRF			
Method	PSNR↑	SSIM↑	LPIPS↓	Size (MB)↓	Method	PSNR↑	SSIM↑	LPIPS↓	Size (MB)↓
PCGS (30k)	30.061	0.9059	0.2624	4.16	PCGS (40k)	26.838	0.8645	0.2203	14.00
	30.279	0.9092	0.2578	5.57		27.075	0.8709	0.2134	18.56
	30.303	0.9097	0.2570	7.25		27.129	0.8730	0.2104	23.44
	30.308	0.9098	0.2567	9.03		27.148	0.8740	0.2086	28.53
	30.306	0.9098	0.2566	10.92		27.153	0.8744	0.2076	33.76
HAC	30.029	0.906	0.266	4.31	HAC	26.483	0.847	0.248	18.58
	30.157	0.907	0.262	4.68		26.685	0.855	0.238	20.35
	30.217	0.909	0.255	6.61		26.833	0.862	0.228	22.72
						27.094	0.871	0.213	26.58
						27.092	0.874	0.207	29.93
HAC++	30.263	0.909	0.260	3.99	HAC++	26.793	0.858	0.234	11.87
	30.349	0.910	0.255	5.23		26.950	0.866	0.223	13.61
	30.325	0.910	0.253	6.73		27.065	0.872	0.211	16.17
						27.218	0.879	0.197	20.68
						27.195	0.882	0.188	25.20
ContextGS	30.117	0.908	0.263	3.78	ContextGS	26.931	0.866	0.222	13.76
	30.253	0.909	0.260	4.38		27.051	0.870	0.217	15.39
	30.164	0.910	0.256	5.51		27.098	0.873	0.211	17.71
	30.199	0.910	0.255	6.75		27.198	0.876	0.205	21.61
						27.151	0.876	0.201	25.58
GoDe	30.150	0.904	0.343	5.40					
	30.290	0.908	0.334	7.20					
	30.330	0.909	0.331	8.40					
HEMGS	30.240	0.909	0.270	2.85					
	30.400	0.912	0.258	6.39					
CAT-3DGS	30.160	0.906	0.275	2.82	CAT-3DGS	26.650	0.859	0.226	16.21
	30.290	0.909	0.269	3.56		27.100	0.875	0.204	20.14
						27.220	0.880	0.195	22.60
						27.350	0.886	0.184	26.59
RACE	30.135	0.9068	0.2638	3.93	RACE	26.993	0.8732	0.2069	15.24
	30.294	0.9091	0.2603	5.03		27.185	0.8783	0.2013	19.97
	30.333	0.9095	0.2593	6.51		27.216	0.8795	0.1993	25.53

Rows within each method are ordered by increasing bitstream size. The new ablation averages 2 Deep Blending scenes and 6 BungeeNeRF scenes. Sizes are decimal MB.

Table 6: Compression rank computed from the selected operating points in Table 1. Lower is better. Rank averages the available dataset contributions; Coverage reports how many of the four datasets are available.

Method	Mip-NeRF 360	Tanks & Temples	Deep Blending	BungeeNeRF	Rank	Coverage
RACE	3.833	3.833	3.500	2.000	**3.292**	4/4
HAC++	3.417	3.667	4.000	3.000	3.521	4/4
HEMGS	3.583	3.500	3.750	–	3.611	3/4
CAT-3DGS	4.667	3.833	3.583	5.167	4.313	4/4
ContextGS	6.167	4.833	4.667	3.000	4.667	4/4
ProGS	4.917	4.917	–	5.333	5.056	3/4
PCGS	7.000	7.167	4.500	2.667	5.333	4/4
HAC	5.750	5.250	5.417	6.833	5.813	4/4
GoDe	5.667	8.000	6.583	–	6.750	3/4

receive their average rank. The quality metrics jointly receive half of the weight, and model size receives the other half:

$$R_{d,m} = \frac{r_{d,m}^{\mathrm{PSNR}}}{6} + \frac{r_{d,m}^{\mathrm{SSIM}}}{6} + \frac{r_{d,m}^{\mathrm{LPIPS}}}{6} + \frac{r_{d,m}^{\mathrm{Size}}}{2}, \tag{12}$$

where $r_{d,m}^{(\cdot)}$ is the rank of method m on dataset d for the indicated metric. Each method contributes exactly one selected real operating point per available dataset from Tables 4 and 5. The overall score averages the available dataset contributions; methods with incomplete coverage are explicitly marked. Table 6 reports the per-dataset and average scores.

C Additional Qualitative Results

Figures 7 and 8 collect the remaining scenes that yield informative local comparisons, using the same size-aligned operating points and GT-guided visualization protocol as Fig. 4.

D Additional Ablations

Fig. 5 summarizes the three component-wise comparisons over all 19 scenes. Fig. 9 reports the dataset-wise PSNR view, while Figs. 10 and 11 provide the corresponding SSIM and LPIPS results. The total compressed model size is averaged at the corresponding progressive operating point.

E Additional Rate-Granularity Analysis

Fig. 6(a) uses granularities emitted by the selector rather than post-hoc choices. Its four scenes span indoor and outdoor content and both three- and four-level protocols. In Fig. 6(b), each point compares identical feature and scaling symbols before and after arithmetic coding; headers are included only in the actual payload. MAPE is computed separately for each scene rather than aggregated across different rate scales.

F Cross-Level Context Analysis

F.1 Progressive Gain at Each Level

Fig. 12 traces the decoded quality against cumulative size at every progressive level, where each point is the PSNR averaged over all test views of the complete scene. We select two representative scenes from each of the four datasets, for a total of eight scenes. PSNR increases monotonically for all eight representative scenes. The effect persists across all four datasets and scene type: the final enhancement over the base layer ranges from 0.061 to 0.309 dB across the tested outdoor and indoor scenes. Thus, the proposed context is useful when decoding successive refinement symbols rather than only improving the final operating point.

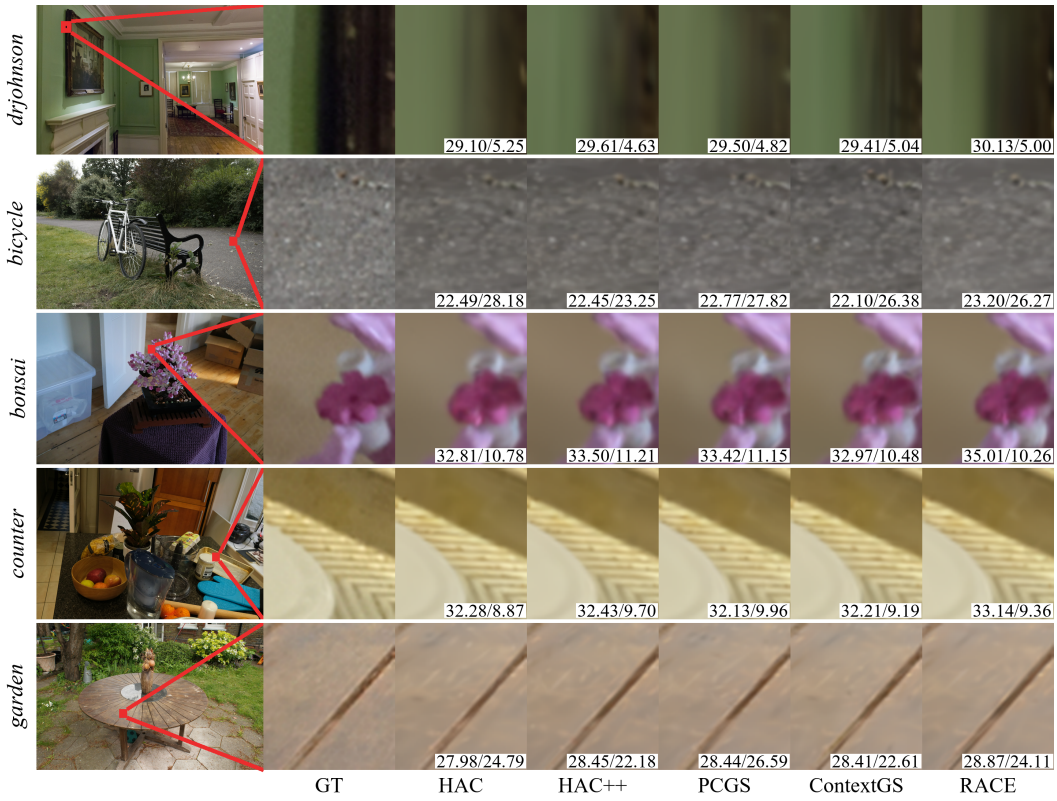


Figure 7: Additional qualitative comparisons (top to bottom: drjohnson, bicycle, bonsai, counter, and garden). The left image in each row shows the ground-truth view and selected region; subsequent columns compare the enlarged crop from GT, HAC, HAC++, PCGS, ContextGS, and RACE. Each inset label reports PSNR [dB] / total compressed model size [MB]. Method names are shown once below the final row.

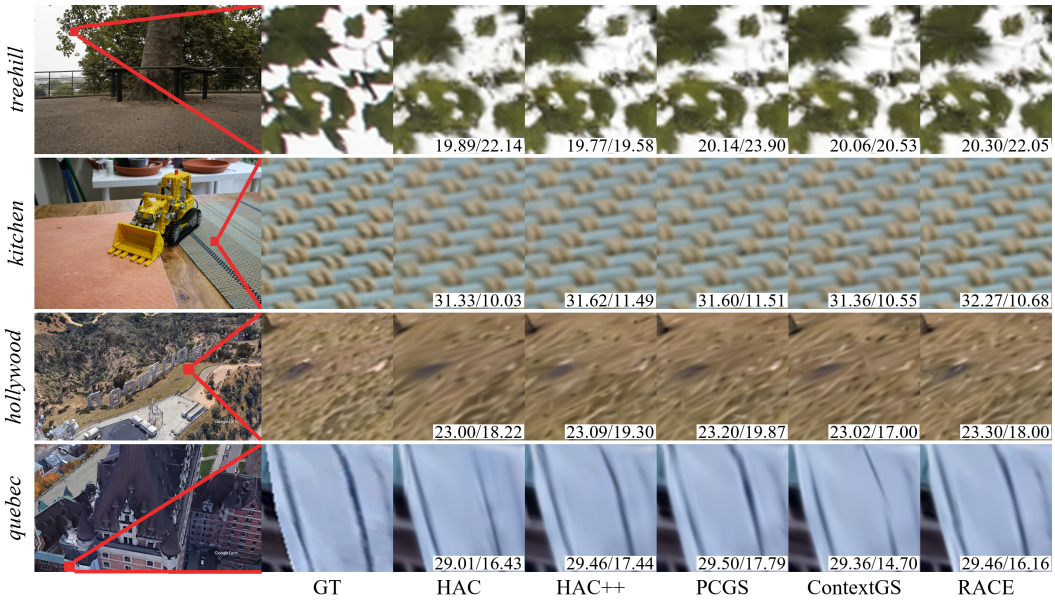


Figure 8: Additional qualitative comparisons (top to bottom: treehill, kitchen, hollywood, and quebec). The visualization protocol and reported quantities follow Fig. 7.

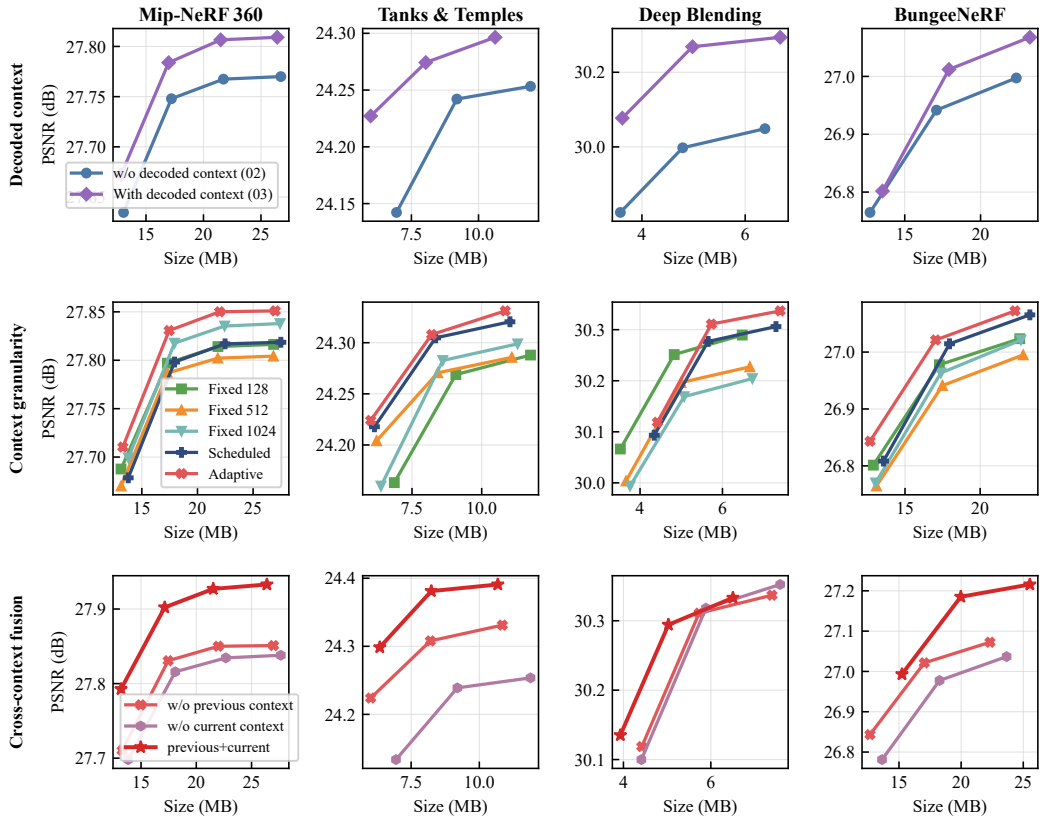


Figure 9: Complete dataset-wise PSNR–size ablations organized by the three components. Rows denote controlled ablation groups and columns denote datasets; each row shows its legend only in the first panel.

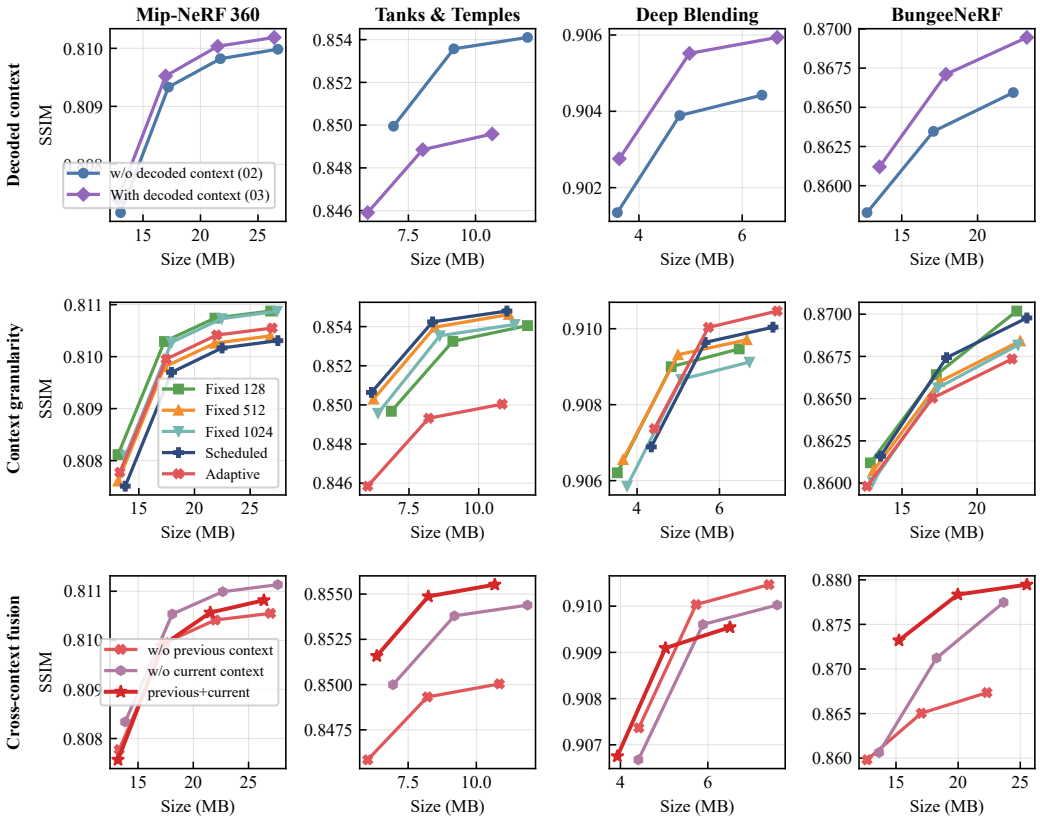


Figure 10: Complete SSIM–size ablations using the same controlled groups and dataset-level averaging protocol as Fig. 5.

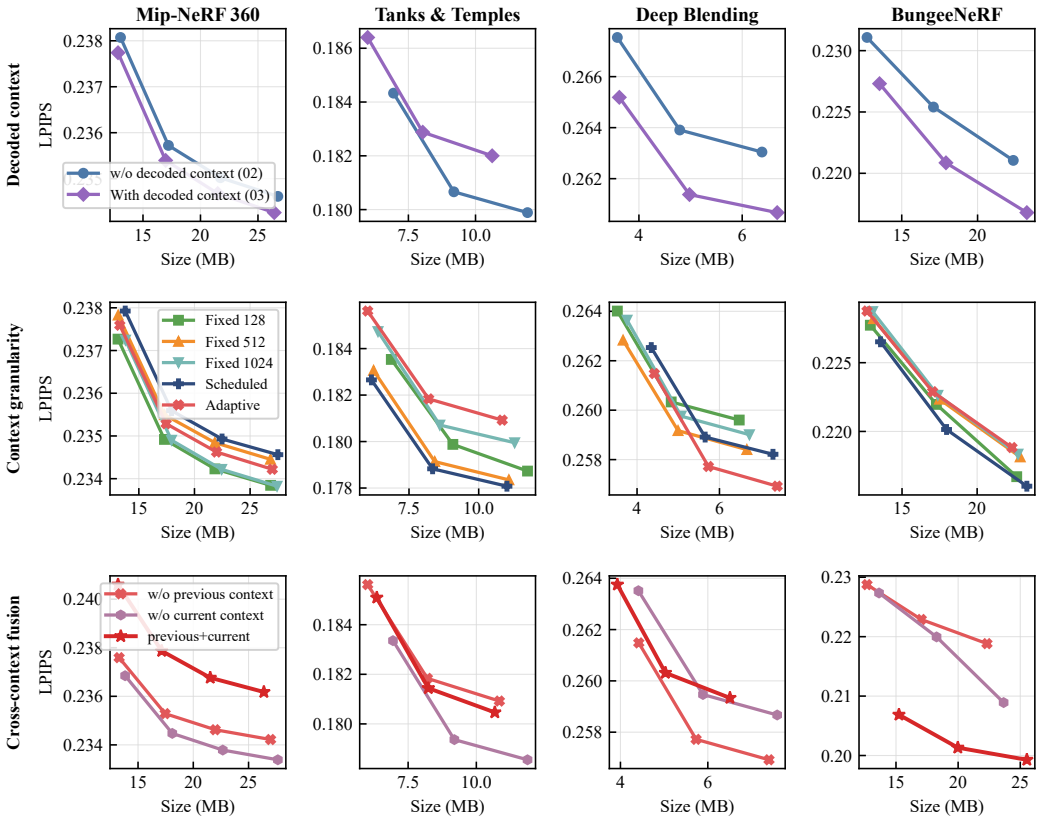


Figure 11: Complete LPIPS–size ablations; lower LPIPS is better. The disaggregated plots confirm whether conclusions drawn from PSNR persist for perceptual quality.

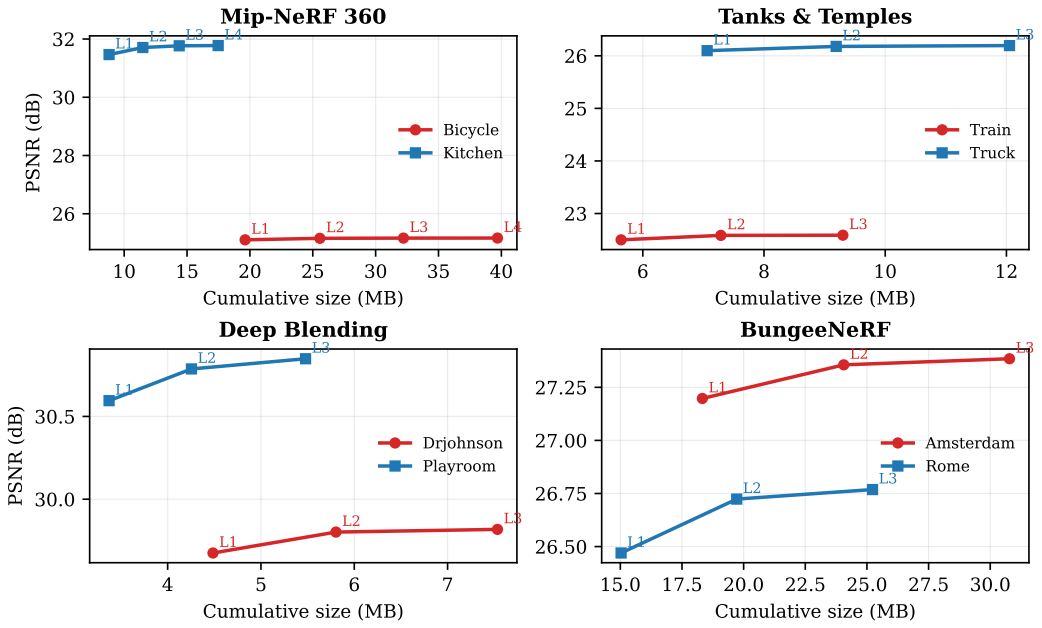


Figure 12: Scene-level average PSNR versus cumulative bitstream size during progressive decoding. We select two representative scenes from each of the four datasets. Each point averages PSNR over all test views of the corresponding scene, and labels L1–L4 denote the available successively decoded levels (three or four according to the dataset protocol). Every scene improves monotonically as refinement layers are received.

F.2 Anchor Symbol Accounting

Table 7 reports, per dataset, the payload share and the per-anchor cost of newly activated versus refined anchor symbols in the complete model. Per-scene values are averaged over refinement levels and then over the scenes of each dataset, and the Average row weights the four datasets equally. Fig. 13(a) plots the two cost columns with their quotient annotated, and Fig. 13(b) gives the share of each refinement level's payload that belongs to anchors activated at that level.

Tables 8–10 expand those averages into the quantities they are computed from, for every scene and every progressive level: the anchor count N, the transmitted mask m_L and its increment $\Delta m = m_L - m_{L-1}$, the implied counts $n_{\text{new}} = N\Delta m$ and $n_{\text{ref}} = N m_L$, the payload of each class, and the derived per-anchor costs. Payloads are reported in MB as emitted by the encoder, and per-anchor costs are obtained as $\text{payload} \times 2^{20}/n$. Level 1 is listed for completeness but excluded from the averages, since every anchor is new there by construction and the level is dominated by the base context header. Fig. 14 plots the same per-scene quantities, sorted by the cost ratio.

Table 7: Payload and per-anchor cost of newly activated versus refined anchor symbols in the complete model (RACE), averaged per dataset. Per-scene values are averaged over refinement levels and then over the scenes of each dataset; the Average row weights the four datasets equally. The ratio is the quotient of the two cost columns. The two classes carry different numbers of scalars per anchor, so the ratio measures the cost per anchor rather than the coding efficiency of a single scalar.

Dataset	Scenes	New share (%)	Bytes / new	Bytes / refined	Ratio ↓
Mip-NeRF 360	9	6.36	47.7	10.7	4.46×
Tanks & Temples	2	5.24	38.1	9.7	3.91×
Deep Blending	2	6.69	31.3	9.7	3.23×
BungeeNeRF	6	6.01	52.4	10.6	4.96×
Average	19	6.07	42.4	10.2	4.16×

Table 8: Per-scene, per-level anchor accounting on Mip-NeRF 360. Column definitions follow Table 7; "–" marks level_1, where no refined anchor exists.

Scene	L	N	m_L	Δm	n_{new}	n_{ref}	new (MB)	ref (MB)	new (%)	B / new	B / ref	ratio
Mip-NeRF 360												
Bicycle	1	883531	0.712762	0.712762	629747	0	16.6255	0.0000	100.00	27.68	0.00	–
	2	883531	0.715562	0.002800	2474	632221	0.1000	6.0042	1.64	42.37	9.96	4.26×
	3	883531	0.723929	0.008367	7393	639614	0.3250	6.5086	4.76	46.10	10.67	4.32×
	4	883531	0.741629	0.017700	15638	655252	0.8065	6.8974	10.47	54.08	11.04	4.90×
Bonsai	1	456631	0.588328	0.588328	268649	0	6.2294	0.0000	100.00	24.31	0.00	–
	2	456631	0.593186	0.004858	2218	270867	0.0715	2.5113	2.77	33.80	9.72	3.48×
	3	456631	0.605044	0.011858	5415	276282	0.2132	2.7707	7.15	41.29	10.52	3.93×
	4	456631	0.622097	0.017053	7787	284069	0.3815	2.9819	11.34	51.37	11.01	4.67×
Counter	1	305205	0.691129	0.691129	210936	0	6.0630	0.0000	100.00	30.14	0.00	–
	2	305205	0.695516	0.004387	1339	212275	0.0565	2.1295	2.59	44.27	10.52	4.21×
	3	305205	0.706397	0.010881	3321	215596	0.1567	2.2333	6.55	49.47	10.86	4.55×
	4	305205	0.720653	0.014256	4351	219947	0.2523	2.3162	9.82	60.81	11.04	5.51×
Flowers	1	704101	0.740787	0.740787	521589	0	14.0802	0.0000	100.00	28.31	0.00	–
	2	704101	0.743926	0.003139	2210	523799	0.0857	5.1440	1.64	40.66	10.30	3.95×
	3	704101	0.753659	0.009733	6853	530652	0.2825	5.5639	4.83	43.23	10.99	3.93×
	4	704101	0.772300	0.018641	13125	543777	0.6659	5.8358	10.24	53.20	11.25	4.73×
Garden	1	764157	0.741068	0.741068	566292	0	16.1496	0.0000	100.00	29.90	0.00	–
	2	764157	0.744703	0.003635	2778	569070	0.1239	5.6266	2.15	46.75	10.37	4.51×
	3	764157	0.754621	0.009918	7579	576649	0.3660	5.9549	5.79	50.63	10.83	4.68×
	4	764157	0.770373	0.015752	12037	588686	0.6904	6.2028	10.02	60.14	11.05	5.44×
Kitchen	1	349603	0.727806	0.727806	254443	0	6.6125	0.0000	100.00	27.25	0.00	–
	2	349603	0.732554	0.004748	1660	256103	0.0724	2.6031	2.71	45.73	10.66	4.29×
	3	349603	0.741882	0.009328	3261	259364	0.1647	2.7111	5.73	52.96	10.96	4.83×
	4	349603	0.755325	0.013443	4700	264064	0.2938	2.7853	9.54	65.54	11.06	5.93×
Room	1	297873	0.582611	0.582611	173544	0	3.9195	0.0000	100.00	23.68	0.00	–
	2	297873	0.592961	0.010350	3083	176627	0.1013	1.6768	5.70	34.45	9.95	3.46×
	3	297873	0.611344	0.018383	5476	182103	0.2107	1.8414	10.27	40.34	10.60	3.80×
	4	297873	0.634673	0.023329	6949	189052	0.3196	1.9732	13.94	48.23	10.94	4.41×
Stump	1	681840	0.550002	0.550002	375013	0	12.1026	0.0000	100.00	33.84	0.00	–
	2	681840	0.551942	0.001940	1323	376336	0.0532	3.6678	1.43	42.15	10.22	4.12×
	3	681840	0.558282	0.006340	4323	380659	0.1851	3.8901	4.54	44.91	10.72	4.19×
	4	681840	0.573066	0.014784	10080	390739	0.4969	4.0749	10.87	51.69	10.94	4.73×
Treehill	1	758309	0.725479	0.725479	550137	0	14.1629	0.0000	100.00	26.99	0.00	–
	2	758309	0.728608	0.003129	2373	552510	0.0940	5.4591	1.69	41.55	10.36	4.01×
	3	758309	0.736957	0.008349	6331	558841	0.2770	5.8572	4.51	45.87	10.99	4.17×
	4	758309	0.751930	0.014973	11354	570195	0.6000	6.1197	8.93	55.41	11.25	4.92×

Table 9: Per-scene, per-level anchor accounting on Tanks & Temples and Deep Blending. Notation follows Table 7.

Scene	L	N	m_L	Δm	n_{new}	n_{ref}	new (MB)	ref (MB)	new (%)	B / new	B / ref	ratio
Tanks & Temples												
Train	1	221672	0.787303	0.787303	174523	0	3.9445	0.0000	100.00	23.70	0.00	–
	2	221672	0.792464	0.005161	1144	175667	0.0534	1.5931	3.25	48.97	9.51	5.15×
	3	221672	0.809827	0.017363	3849	179516	0.1689	1.8440	8.39	46.01	10.77	4.27×
Truck	1	381699	0.681451	0.681451	260109	0	5.1212	0.0000	100.00	20.64	0.00	–
	2	381699	0.686651	0.005200	1985	262094	0.0517	2.0778	2.43	27.29	8.31	3.28×
	3	381699	0.704605	0.017954	6853	268947	0.1968	2.6630	6.88	30.12	10.38	2.90×
Deep Blending												
Drjohnson	1	222317	0.664659	0.664659	147765	0	2.8318	0.0000	100.00	20.09	0.00	–
	2	222317	0.669787	0.005128	1140	148905	0.0352	1.2828	2.67	32.37	9.03	3.58×
	3	222317	0.691396	0.021609	4804	153709	0.1655	1.5656	9.56	36.13	10.68	3.38×
Playroom	1	178518	0.577583	0.577583	103109	0	1.7997	0.0000	100.00	18.30	0.00	–
	2	178518	0.584518	0.006935	1238	104347	0.0312	0.8519	3.54	26.47	8.56	3.09×
	3	178518	0.610600	0.026082	4656	109003	0.1342	1.0885	10.97	30.22	10.47	2.89×

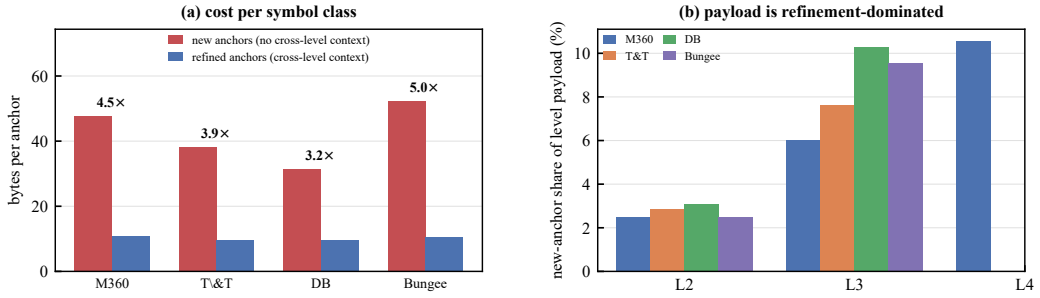


Figure 13: Payload accounting of the complete model. (a) Cost per anchor of the two symbol classes, averaged per dataset; the annotated factor is the quotient of the two bars. (b) Share of each refinement level's payload that belongs to anchors activated at that level. Every level is refinement dominated, and the class that can be predicted from a decoded cross-level state is consistently the cheaper one. Detailed results are given in Appendix F.2.

Table 10: Per-scene, per-level anchor accounting on BungeeNeRF. Notation follows Table 7.

Scene	L	N	m_L	Δm	n_{new}	n_{ref}	new (MB)	ref (MB)	new (%)	B / new	B / ref	ratio
BungeeNeRF												
Amsterdam	1	684738	0.821387	0.821387	562435	0	15.1259	0.0000	100.00	28.20	0.00	–
	2	684738	0.826256	0.004869	3334	565769	0.1498	5.5828	2.61	47.12	10.35	4.55×
	3	684738	0.846477	0.020221	13846	579615	0.7120	6.0103	10.59	53.92	10.87	4.96×
Bilbao	1	513605	0.849838	0.849838	436481	0	11.4916	0.0000	100.00	27.61	0.00	–
	2	513605	0.854982	0.005144	2642	439123	0.1252	4.3330	2.81	49.68	10.35	4.80×
	3	513605	0.876136	0.021154	10865	449988	0.5877	4.6725	11.17	56.72	10.89	5.21×
Hollywood	1	460139	0.893232	0.893232	411011	0	11.6748	0.0000	100.00	29.78	0.00	–
	2	460139	0.896112	0.002880	1325	412336	0.0847	4.0963	2.02	67.00	10.42	6.43×
	3	460139	0.908499	0.012387	5700	418036	0.3598	4.3817	7.59	66.19	10.99	6.02×
Pompidou	1	670461	0.788253	0.788253	528493	0	13.5537	0.0000	100.00	26.89	0.00	–
	2	670461	0.793541	0.005288	3545	532038	0.1500	5.1186	2.85	44.35	10.09	4.40×
	3	670461	0.815868	0.022327	14969	547008	0.6764	5.6277	10.73	47.38	10.79	4.39×
Quebec	1	493376	0.834441	0.834441	411693	0	10.2559	0.0000	100.00	26.12	0.00	–
	2	493376	0.838944	0.004503	2222	413915	0.0931	3.9938	2.28	43.93	10.12	4.34×
	3	493376	0.855794	0.016850	8313	422228	0.3976	4.3909	8.30	50.14	10.90	4.60×
Rome	1	552320	0.852385	0.852385	470789	0	12.0093	0.0000	100.00	26.75	0.00	–
	2	552320	0.856661	0.004276	2362	473151	0.1089	4.5798	2.32	48.35	10.15	4.76×
	3	552320	0.873859	0.017198	9499	482650	0.4858	5.0215	8.82	53.63	10.91	4.92×

F.3 Estimation of the CAT-3DGS Bitstream Allocation

CAT-3DGS (Zhan et al., 2025a) reports scene-wise total compressed sizes but does not report the corresponding component-wise allocation. We therefore use the total size from the original paper and estimate only its decomposition from our locally encoded bitstreams. For the same scene and rate parameter, let S^{paper} denote the reported total size and s_i^{local} the locally measured size of mutually exclusive component i. The displayed component size is

$$\widehat{s}_i = S^{\text{paper}} \frac{s_i^{\text{local}}}{\sum_j s_j^{\text{local}}}. \tag{13}$$

The normalization makes the estimated components sum exactly to the reported total size and removes small discrepancies caused by logging precision. A parent component and its children are never accumulated simultaneously; for example, the total feature payload is used in the top-level decomposition, whereas its level-wise payloads are shown only as an internal subdivision. Consequently, CAT-3DGS totals and rate–distortion points remain the values reported by its authors, while only the component allocation is estimated. The local encoder completed all component streams before the subsequent decoding error, so these streams are used solely to obtain relative allocation ratios. They are not presented as successful end-to-end reconstructions.

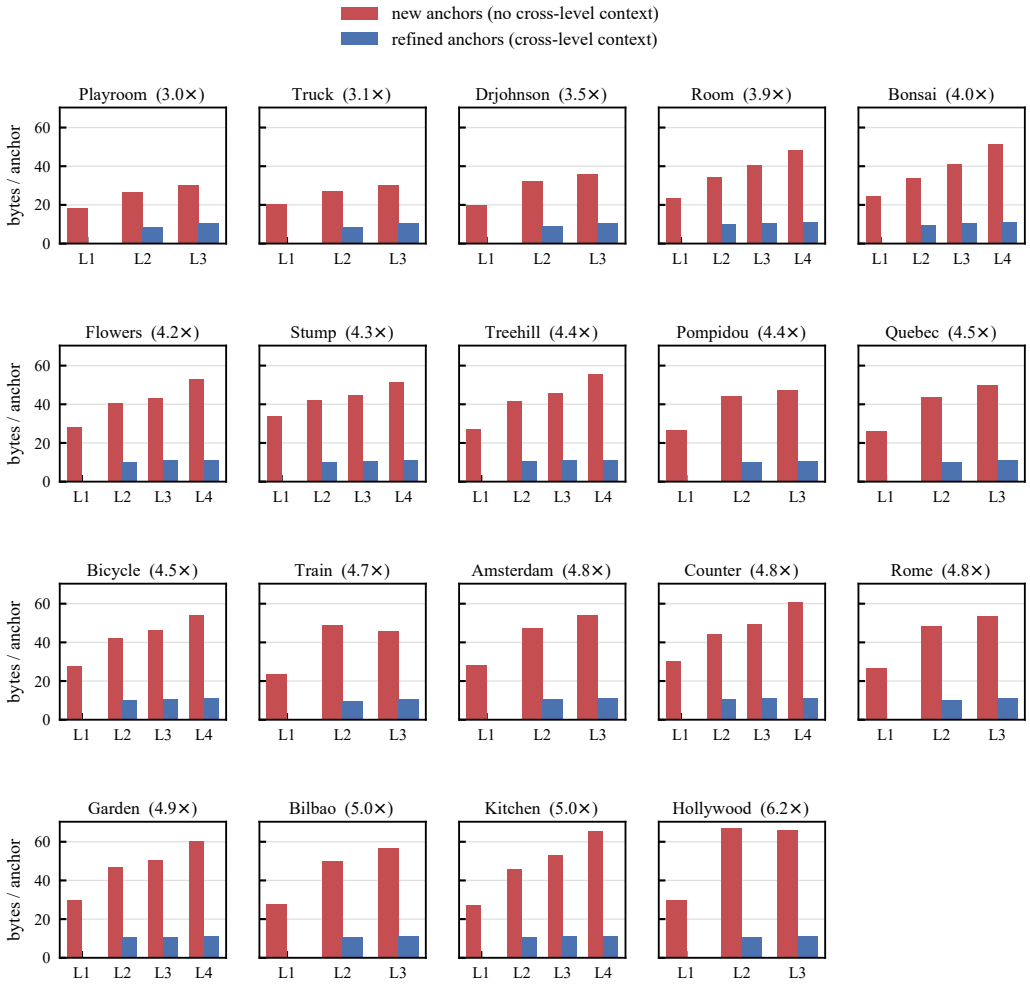


Figure 14: Per-anchor cost of the two symbol classes for all 19 scenes of the complete model, one panel per scene and one bar pair per progressive level. Panels are sorted by the mean cost ratio of the scene, reported in each title.

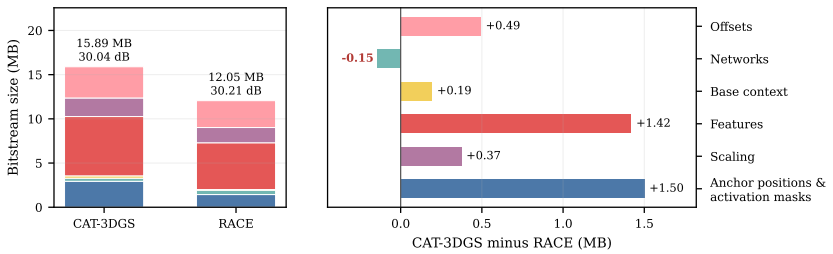


Figure 15: Context cost and coding benefit on Quebec. Left: bitstream composition. Right: CAT-3DGS component size minus that of RACE.

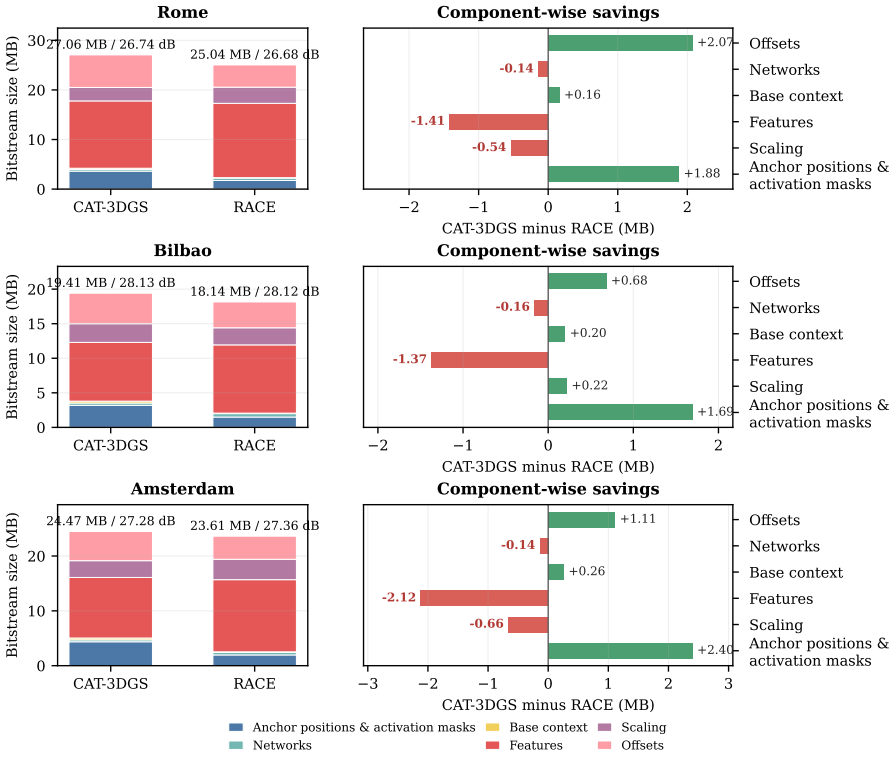


Figure 16: Supplementary RACE–CAT-3DGS comparisons on Rome, Bilbao, and Amsterdam, following the layout of Fig. 15(b). Each left panel shows the closest available quality points and their component-wise sizes; each right panel reports CAT-3DGS minus RACE. The CAT-3DGS decomposition is estimated by Eq. (13), whereas RACE components are measured directly.

Fig. 16 and Table 11 supplement the Quebec comparison in the main paper with Rome, Bilbao, and Amsterdam. RACE adds only 0.14–0.16 MB of entropy-network parameters, while reducing the total size by 0.86–2.02 MB at closely matched PSNR. In particular, its savings on anchor positions and masks reach 1.69–2.40 MB, and its offset savings reach 0.68–2.07 MB. Some scenes allocate more bits to features or scaling, showing that the gain is a redistribution driven by improved probability modeling rather than a uniform reduction of every stream. These component comparisons do not replace the rate–distortion measurements reported by the CAT-3DGS authors.

Table 11: Supplementary component-wise comparison at the closest available quality points. CAT-3DGS totals and quality are taken from its paper, and its components are estimated by Eq. (13); RACE values are measured. Sizes are in MB.

Scene	Method	PSNR	Total	Anchor+mask	Networks	Base context	Features	Scaling	Offsets
Rome	CAT-3DGS	26.740	27.060	3.621	0.352	0.236	13.577	2.721	6.554
	RACE	26.681	25.035	1.740	0.493	0.074	14.992	3.257	4.480
Bilbao	CAT-3DGS	28.130	19.410	3.191	0.332	0.273	8.504	2.694	4.415
	RACE	28.121	18.143	1.496	0.493	0.074	9.873	2.474	3.733
Amsterdam	CAT-3DGS	27.280	24.470	4.357	0.355	0.336	11.041	3.057	5.325
	RACE	27.358	23.615	1.958	0.493	0.074	13.159	3.719	4.212