HHC-GS: HYPERBOLIC HIERARCHICAL CONTEXT FOR COMPACT MULTISCALE GAUSSIAN SPLATTING

Anonymous authors
Paper under double-blind review

ABSTRACT

3D Gaussian Splatting (3DGS) enables high-quality rendering with explicit scene representations, but incurs substantial storage costs in large-scale scenes. Existing multiscale methods reduce rendering overhead by selectively activating anchors at different levels of detail, while retaining independently stored high-dimensional features for individual anchors. To reduce this redundancy, we propose HHC-GS, a compact multiscale Gaussian mapping method based on hyperbolic hierarchical context. HHC-GS uses hyperbolic relations to modulate ancestor-conditioned feature prediction, allowing each non-root anchor to store only low-dimensional structural variables and a bounded residual. For local map loading, feature-generation dependencies are restricted to ancestor paths, and dependency closure ensures consistent feature generation between local subsets and the complete map. We further jointly optimize multiple hierarchy prefixes with frozen-teacher supervision to improve reconstruction under different representation budgets while preserving full-map quality. Experiments on MatrixCity, BungeeNeRF, and UrbanScene3D show that HHC-GS maintains competitive reconstruction quality across representation budgets while substantially reducing storage. Compared with Octree-GS, HHC-GS achieves comparable or better reconstruction quality while reducing model size by more than 85% on all three datasets.

1 INTRODUCTION

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) achieves high-quality real-time rendering through explicit Gaussian primitives and differentiable rasterization, and has increasingly been extended to urban scene reconstruction (Lin et al., 2024; Liu et al., 2025b;c). Compared with indoor environments, urban scenes span much larger spatial extents (Liu et al., 2025a; Yuan et al., 2025), where nearby details and distant structures require different levels of detail (Yu et al., 2024). This motivates multiscale representations, but storing information across spatial regions and hierarchy levels introduces substantial map storage costs (Kerbl et al., 2024). Moreover, limited runtime memory often motivates partial map loading, while the loaded subset must still support effective reconstruction (Shi et al., 2025). Thus, large-scale Gaussian mapping requires both compact multiscale storage and reliable reconstruction under partial loading.

Existing multiscale Gaussian representations mainly follow three directions: (i) hierarchical methods organize anchors into multiple levels and select levels of detail according to viewing distance (Ren et al., 2025; Lu et al., 2024); (ii) cross-level context methods exploit correlations between hierarchy levels for feature enhancement or conditional modeling (Zhang et al., 2026; Wang et al., 2024); and (iii) progressive representations use hierarchy prefixes to accommodate different resource budgets (Guo et al., 2026; Zheng et al., 2025; Tang et al., 2026). While these approaches improve multiscale rendering and resource adaptability, they generally retain independently stored node features with relatively high dimensionality. This leaves substantial redundancy across hierarchically related nodes and limits the storage efficiency of the representation.

Reducing this redundancy introduces three challenges. First, hierarchical context should reduce the information stored independently at each node, rather than only refine features of unchanged dimensionality. Second, feature generation should avoid dependencies on nodes outside the loaded local region, which would otherwise introduce additional loading overhead (Windisch et al., 2026). Third, reconstruction across hierarchy levels must be coordinated because Gaussians from different

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.

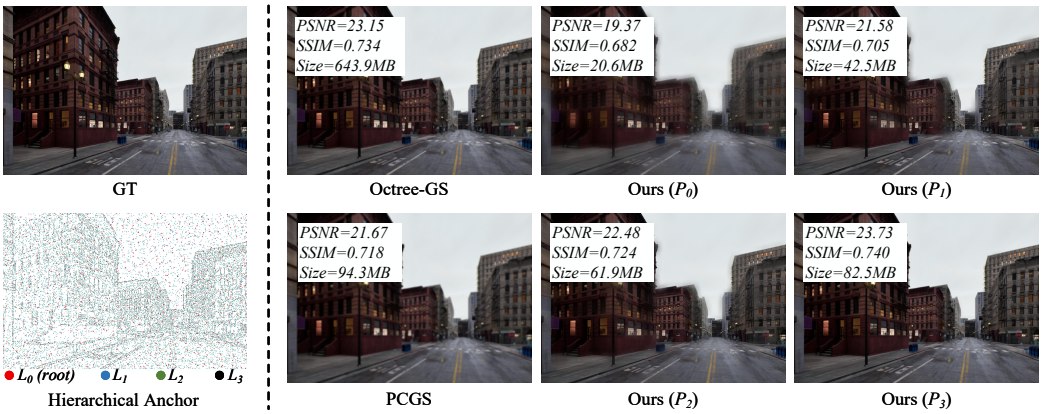


Figure 1: Overview and qualitative comparison of HHC-GS on the MatrixCity Small City street. The hierarchical anchors contain four levels (L_0–L_3), forming prefixes P_0–P_3, enabling progressive reconstruction across representation budgets. Using the full hierarchy prefix P_3, HHC-GS reduces storage by 87.2% compared with Octree-GS while improving PSNR by 0.58 dB.

levels jointly contribute through alpha compositing, and fine-level representations are not simple image-space residuals of coarse-level renderings (Chen et al., 2026; Seo et al., 2025). To address these issues, we exploit ancestor paths in the existing spatial tree as structured context. Since hyperbolic space naturally represents branching hierarchies with low-dimensional embeddings (Nickel & Kiela, 2017; 2018; Sala et al., 2018), we use hyperbolic relations to modulate the contribution of ancestor information.

Based on this design, we propose HHC-GS, a compact multiscale Gaussian mapping framework based on hyperbolic hierarchical context. As illustrated in Fig. 1, HHC-GS supports progressive reconstruction across hierarchy prefixes with different representation sizes. HHC-GS preserves the existing spatial tree topology and generates non-root anchor features from ancestor context, while each node stores only low-dimensional structural variables and a bounded residual. Feature dependencies are restricted to ancestor paths within each block, and ancestor-complete loading ensures that local prefixes follow the same generation rule as the complete map. To maintain reconstruction quality across loaded-representation budgets, we jointly optimize multiple hierarchy prefixes and the complete representation, together with a filtered consistency constraint derived from a frozen teacher.

Our contributions are as follows:

- We propose a compact conditional Gaussian representation that combines hyperbolic ancestor modulation with low-dimensional bounded residuals, reducing the node-specific information that must be stored independently.
- We introduce branch-aware ancestor-conditioned feature generation and block-level dependency closure, enabling local prefixes to reproduce the same feature-generation process as the complete map without introducing dependencies outside the loaded region.
- We develop a budget-aware joint reconstruction strategy for varying loaded representation sizes, using multi-prefix supervision and frozen-teacher consistency to improve coarse-level reconstruction while preserving complete-map quality.

2 Related Work

2.1 Multiscale Gaussian Scene Representations

3DGS (Kerbl et al., 2023) represents scenes with explicit Gaussian primitives. Scaffold-GS (Lu et al., 2024) introduces anchor-based Gaussian generation, providing a basis for hierarchical scene representations. Octree-GS (Ren et al., 2025) organizes anchors in an octree and selects levels

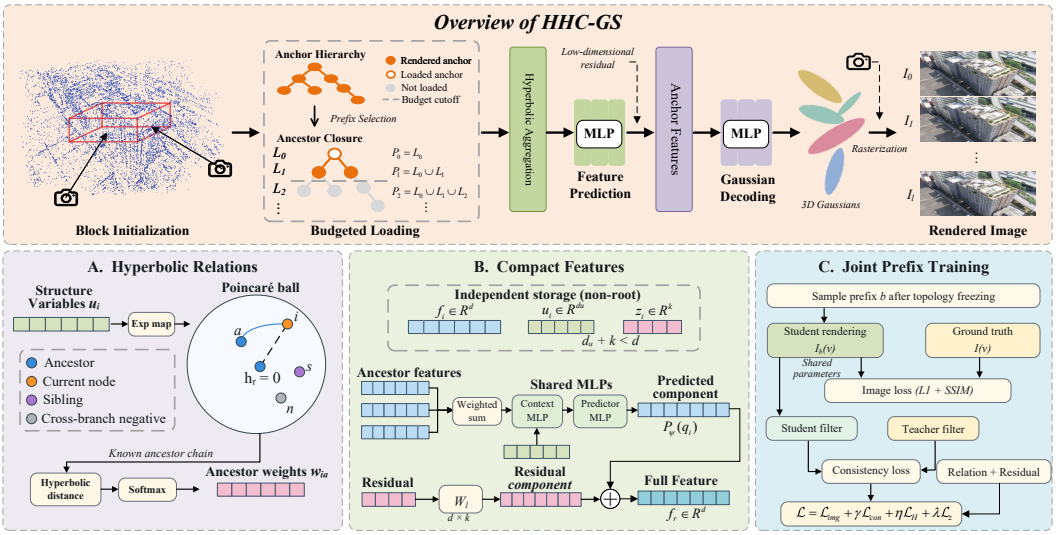


Figure 2: Overview of HHC-GS for compact multiscale Gaussian scene representation. The framework combines hyperbolic ancestor weighting, compact feature generation with low-dimensional residuals, and joint prefix training with teacher-guided consistency. Ancestor-closed loading supports local feature generation, while a shared Gaussian decoder enables rendering under different loaded-representation budgets.

according to viewing distance. HiCo-GS (Zhang et al., 2026) further exploits cross-level context for feature aggregation. LapisGS (Shi et al., 2025) and MGS (Guo et al., 2026) support resource-adaptive rendering through progressive representations and multi-prefix optimization, respectively. These methods mainly improve level-of-detail selection, cross-level interaction, or progressive reconstruction. In contrast, we use ancestor context to generate node features while reducing the dimensionality of independently stored variables, enabling compact reconstruction across hierarchy prefixes.

2.2 Compact Gaussian Representations and Hierarchical Modeling

Existing 3DGS compression methods reduce storage through primitive reduction (Fang & Wang, 2024; Hanson et al., 2025; Wang et al., 2025), compact attribute representations (Lee et al., 2024; Shin et al., 2025), and quantization or coding (Niedermayr et al., 2024; Navaneet et al., 2025; Morgenstern et al., 2025). LightGaussian (Fan et al., 2024) combines pruning, attribute simplification, and quantization. HAC (Chen et al., 2025a) and HAC++ (Chen et al., 2025b) exploit spatial context for conditional entropy modeling, while ContextGS (Wang et al., 2024) predicts fine-level attribute distributions from coarse-level anchors. Unlike these context-based compression methods, we use hierarchical context to generate complete node features while storing only low-dimensional node-specific variables.

Hyperbolic representations provide compact embeddings for hierarchical structures. Poincaré embeddings (Nickel & Kiela, 2017), the Lorentz model (Nickel & Kiela, 2018), and subsequent hyperbolic graph models (Chami et al., 2019; Gulcehre et al., 2018) demonstrate their effectiveness for hierarchical and branching data (Sala et al., 2018). Motivated by this property, we use hyperbolic distances to modulate information sharing along known ancestor chains, while preserving the original spatial tree topology and Euclidean Gaussian representation.

3 Method

3.1 Preliminaries

We consider static scene reconstruction and novel view synthesis from calibrated images with known camera poses, using an anchor-based Gaussian representation (Ren et al., 2025). The center $x_i \in \mathbb{R}^3$

of anchor i is fixed within its associated voxel. Its feature $f_i \in \mathbb{R}^d$ and non-feature attributes $a_i \in \mathbb{R}^{d_a}$ jointly generate view-dependent Gaussians. Given a camera v and an active anchor set S, the rendered image is

$$I_S(v) = \mathcal{R}\left(\{\mathcal{D}_\theta(f_i, a_i, v)\}_{i\in S}, v\right), \tag{1}$$

where $\mathcal{D}_\theta$ is the Gaussian attribute decoder, $\mathcal{R}$ is the splatting renderer, and θ denotes the decoder parameters.

We partition the scene into spatial blocks and construct a rooted tree $\mathcal{T}_j = (V_j, E_j)$ within each block j. The block-wise trees form a forest $\mathcal{T} = \{\mathcal{T}_j\}$. We denote the parent of node i by $p(i)$ and its level by $l_i \in \{0, \dots, L\}$. During densification, missing ancestors are inserted according to occupied voxels so that each tree remains parent-closed. Given a maximum available level b, we define the global hierarchy prefix P_b and the active anchor set for camera v as

$$P_b = \{i \in V(\mathcal{T}) : l_i \le b\}, \qquad S_b(v) = \mathcal{F}_v\left(\{i \in V(\mathcal{T}) : l_i \le \min\left[b, \ell_v(i)\right]\}\right), \tag{2}$$

where $V(\mathcal{T}) = \bigcup_j V_j$, $\ell_v(i)$ is the target level determined by the baseline viewing-distance rule, and $\mathcal{F}_v$ performs frustum and support-region filtering. We adopt a cumulative multilevel representation: anchors from multiple levels that satisfy the filtering criterion jointly contribute to alpha compositing, rather than using finer levels to replace coarser ones. The corresponding rendering is denoted by $I_b(v) = I_{S_b(v)}(v)$.

Camera poses remain fixed throughout training. The complete map first undergoes densification and pruning, while ancestors of surviving nodes are retained to preserve the parent-closed hierarchy. Once the topology stabilizes, the tree structure is frozen for compact representation learning and multi-prefix optimization. Fig. 2 summarizes the overall framework, including hyperbolic ancestor modeling, compact conditional feature generation, ancestor-closed budgeted loading, and joint prefix training.

3.2 Hyperbolic Hierarchical Context Mapping

Root nodes directly store full-dimensional features. Each non-root node instead stores a low-dimensional structural variable $u_i \in \mathbb{R}^{d_u}$ and a residual variable $z_i \in \mathbb{R}^k$. We constrain u_i to a bounded region of radius U in the tangent space and map it to the Poincaré ball with curvature $-c$, where $c > 0$:

$$u_i \leftarrow \Pi_U(u_i), \qquad h_i = \operatorname{Exp}_0^c(u_i) = \frac{\tanh\left(\sqrt{c}\,\|u_i\|_2\right)}{\sqrt{c}\,\|u_i\|_2} u_i, \tag{3}$$

where Π_U denotes Euclidean projection onto the radius-U ball in the tangent space. When $\|u_i\|_2 = 0$, the continuous limit is used. For each root node r, we set $u_r = h_r = 0$, while non-root structural variables are learned jointly with the reconstruction objective.

For a non-root node $i \in V_j$, let A_i denote its set of strict ancestors within the same block. Context is aggregated only along this known ancestor chain without relearning the parent–child topology. Hyperbolic distances determine the relative contributions of different ancestors, while the structural and spatial displacements from the parent provide local branch cues:

$$\begin{aligned} w_{ia} &= \frac{\exp\left[-d_c(h_i, h_a)/\tau\right]}{\sum_{r\in A_i} \exp\left[-d_c(h_i, h_r)/\tau\right]}, \qquad a \in A_i, \quad \tau > 0, \\ q_i &= C_\phi\left(\sum_{a\in A_i} w_{ia} f_a,\ u_i - u_{p(i)},\ x_i - x_{p(i)},\ l_i\right), \end{aligned} \tag{4}$$

where τ is a temperature parameter and C_ϕ is a shared context encoder. The displacement $u_i - u_{p(i)}$ lies in the tangent space at the origin, whereas $d_c(\cdot,\cdot)$ is used only for ancestor weighting and does not represent physical scene distance.

Given the ancestor context q_i, a shared predictor generates the main component of the node feature, while the node contributes only a bounded low-dimensional residual:

$$f_i = P_\psi(q_i) + W_{l_i} z_i, \qquad W_l^\top W_l = I_k, \qquad \|z_i\|_2 \le \rho, \qquad d_u + k < d, \tag{5}$$

where P_ψ is a shared predictor and $W_l \in \mathbb{R}^{d\times k}$ is a level-specific basis with orthonormal columns. The orthonormal basis preserves the residual norm, while $d_u + k < d$ ensures that each non-root node stores fewer node-specific feature parameters than a full d-dimensional feature.

Within each block, features are generated recursively in level order, while nodes at the same level can be processed in parallel. Let $f_i^{(b)}$ denote the feature generated for node i when prefix P_b is loaded. Because each non-root node depends only on its strict ancestors and every prefix is parent-closed, fixed parameters and topology imply $f_i^{(b)} = f_i^{(L)}$ for all $i \in P_b$. Thus, a node has the same generated feature in a local prefix and in the complete hierarchy, although the corresponding rendered images may differ because their active anchor sets differ. Generated features may be cached at runtime to avoid repeated recursive evaluation.

3.3 Branch-Aware Relations and Capacity Constraints

The hyperbolic distance used for ancestor weighting in the Poincaré ball with curvature $-c$ is

$$d_c(h_i, h_j) = \frac{1}{\sqrt{c}} \operatorname{arcosh}\left(1 + \frac{2c\|h_i - h_j\|_2^2}{(1 - c\|h_i\|_2^2)\,(1 - c\|h_j\|_2^2)}\right). \tag{6}$$

To encourage the low-dimensional structural variables to preserve both level and branch relations, we introduce a lightweight relation regularizer. For each non-root node, its radial distance is defined as $r_i = d_c(0, h_i)$. A negative sample n is selected from nodes in the same spatial block and at the same depth as i but with a different parent, while a sibling sample s is selected from nodes sharing the parent of i. If a required sample set is empty, the corresponding term is omitted. The relation loss is

$$\begin{aligned}\mathcal{L}_H = {} & \mathbb{E}_{i,n}\left[\max\left(0,\, m_d + d_c(h_i, h_{p(i)}) - d_c(h_i, h_n)\right)\right] \\ & + \beta\, \mathbb{E}_i\left[\max\left(0,\, m_r + r_{p(i)} - r_i\right)\right] + \xi\, \mathbb{E}_{i,s}\left[\max\left(0,\, m_s - d_c(h_i, h_s)\right)\right],\end{aligned} \tag{7}$$

where $m_d, m_r, m_s > 0$ are fixed margins and β, ξ are loss weights. The first term makes the parent closer to the current node than nodes from other branches at the same depth, the second encourages radial distance to increase along the hierarchy, and the third prevents sibling embeddings from collapsing. For a node with only one ancestor, the softmax weight in Eq. 4 is trivially equal to 1; hence, hyperbolic distance-based ancestor reweighting becomes effective only when multiple ancestors are available.

From Eq. 3, the radial distance satisfies $r_i = 2\|u_i\|_2$. Let $\varepsilon \in (0, 1)$ denote a numerical margin from the Poincaré-ball boundary such that $\sqrt{c}\|h_i\|_2 \le 1 - \varepsilon$. To accommodate a radial margin m_r across a hierarchy of maximum level L, we choose U such that $Lm_r \le 2U \le \frac{2}{\sqrt{c}} \operatorname{artanh}(1 - \varepsilon)$. This condition guarantees feasibility of the radial constraint but does not require all sibling and cross-branch margin constraints to be simultaneously satisfiable.

At deployment, root nodes store full-dimensional features, whereas non-root nodes store only (u_i, z_i) together with the Gaussian attributes required for decoding. The map additionally contains spatial indices, tree structures, and shared model parameters. Let N be the total number of anchors, $\mathcal{R}_0$ the root set, and $N_0 = |\mathcal{R}_0|$. If each anchor has d_a non-feature attributes and each scalar occupies q bytes, the complete-map storage is

$$S_{\text{map}} = q\,[N_0 d + (N - N_0)(d_u + k) + N d_a] + S_{\text{idx}} + S_{\text{net}}, \tag{8}$$

where S_{idx} accounts for anchor coordinates, tree structures, and spatial block indices, while S_{net} contains the shared encoder, predictor, and level-specific bases. Under fixed topology and numerical precision, each non-root node reduces its independently stored feature dimensionality by $d - (d_u + k)$. The actual reduction in total map storage also depends on index and shared-model overhead. The deployment storage excludes generated feature caches and optimizer states.

3.4 Budget-Aware Multiscale Reconstruction

Because the representation is cumulative, coarse-level Gaussians remain active when finer levels are introduced and therefore influence subsequent alpha compositing. Moreover, the residual term in Eq. 5 is a feature-space residual rather than an image-space residual. Independently optimizing

different prefixes therefore does not explicitly coordinate their reconstruction quality across loaded-representation budgets. We instead jointly supervise sampled prefixes at a fixed output resolution after the hierarchy topology has been frozen.

For on-demand rendering, we first determine a candidate block set $J(v)$ for camera v using the view frustum and Gaussian support regions. Block bounds are expanded according to the bounded Gaussian offsets and scales so that Gaussians whose support crosses block boundaries are included. For block j, its level-b prefix is $P_{j,b} = \{i \in V_j : l_i \le b\}$. We first collect the view-relevant anchors from the selected block prefixes as

$$\widetilde{C}_b(v) = \bigcup_{j \in J(v)} \left(P_{j,b} \cap S_b(v)\right), \qquad C_b(v) = \mathrm{Anc}\Big(\widetilde{C}_b(v)\Big), \qquad \widetilde{C}_b(v) \subseteq C_b(v), \tag{9}$$

where $\mathrm{Anc}(\cdot)$ denotes ancestor closure within each block-wise tree. Because view-dependent filtering may remove required ancestors, we apply ancestor closure before loading to preserve the same feature-generation rule as the complete map. Closure-only anchors support feature generation but do not participate in rendering.

Let $S_b^{\mathrm{load}}(v)$ denote the amount of representation data associated with the ancestor-closed loading set $C_b(v)$ for rendering camera v with prefix b. Given a loaded-representation budget B, the largest feasible level is

$$b^*(B, v) = \max\left\{b \in \{0, \dots, L\} : S_b^{\mathrm{load}}(v) \le B\right\}. \tag{10}$$

If $S_0^{\mathrm{load}}(v) > B$, no hierarchy prefix is feasible under the given budget. Here, B constrains loaded representation size rather than peak runtime GPU memory.

For all prefix levels, reconstruction is supervised using the same cameras and output resolution. Let $I^{\mathrm{gt}}(v)$ denote the ground-truth image and $|\Omega|$ the number of valid pixels. The reconstruction loss at level b is

$$\mathcal{L}_{\mathrm{img}}^b(v) = (1-\alpha)\frac{\|I_b(v) - I^{\mathrm{gt}}(v)\|_1}{|\Omega|} + \alpha\left[1 - \mathrm{SSIM}\left(I_b(v), I^{\mathrm{gt}}(v)\right)\right]. \tag{11}$$

After complete-map pretraining and topology freezing, we retain a parameter snapshot as a frozen teacher and precompute its full-level renderings $\bar{I}_L(v)$. To transfer coarse-scale scene content from the complete representation to lower prefixes, we use a fixed level-specific low-pass filtering operator G_b and define

$$\mathcal{L}_{\mathrm{con}}^b(v) = \frac{\left\|G_b[I_b(v)] - G_b[\bar{I}_L(v)]\right\|_1}{|\Omega|}. \tag{12}$$

The filtering configuration for each level is fixed throughout training and remains unchanged across controlled comparisons. The teacher receives no gradient updates and is used only during training, so it contributes no deployment storage.

After the topology is frozen, a prefix level b is sampled from a fixed distribution π, with nonzero sampling probabilities assigned to both the root and full levels. We further regularize the non-root residual codes using $\mathcal{L}_z = (N - N_0)^{-1} \sum_{i \notin \mathcal{R}_0} \|z_i\|_2^2$. Let Θ denote all trainable student parameters, including the decoder parameters θ, root features $\{f_r\}_{r \in \mathcal{R}_0}$, Gaussian attributes, non-root structural and residual variables, the context encoder C_ϕ, the predictor P_ψ, and the level-specific bases $\{W_l\}$. The final objective is

$$\min_{\Theta} \; \mathbb{E}_{b \sim \pi,\, v}\left[\mathcal{L}_{\mathrm{img}}^b(v) + \gamma \mathcal{L}_{\mathrm{con}}^b(v)\right] + \eta \mathcal{L}_H + \lambda \mathcal{L}_z. \tag{13}$$

During optimization, u_i and z_i are projected onto their feasible sets to enforce the constraints in Eqs. 3 and 5, while orthogonal retraction maintains $W_l^\top W_l = I_k$.

To compare hierarchy levels under a fixed output resolution, we define the prefix distortion as $D_b = \mathbb{E}_v \mathcal{L}_{\mathrm{img}}^b(v)$. The resource–quality operating points are summarized as

$$\mathcal{P} = \left\{\left(\bar{S}_b^{\mathrm{load}}, D_b, t_b, M_b\right) : b = 0, \dots, L\right\}, \tag{14}$$

where $\bar{S}_b^{\mathrm{load}} = \mathbb{E}_v[S_b^{\mathrm{load}}(v)]$ is the average loaded representation size, t_b is the average end-to-end latency including loading, feature generation, and rendering, and $M_b(v)$ is the peak runtime GPU memory for rendering view v with prefix b, including loaded representation data, generated feature caches, and rendering buffers. We define $M_b = \max_v M_b(v)$. The complete-map storage S_{map} and peak training memory M_{train} are reported separately because they are not prefix-dependent operating variables.

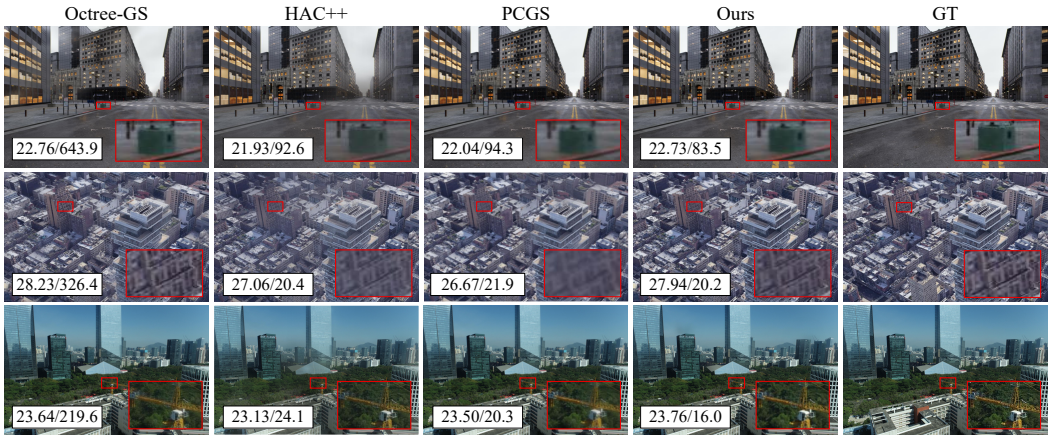


Figure 3: Qualitative comparison on representative scenes from MatrixCity, BungeeNeRF, and UrbanScene3D. PSNR (dB) / Size (MB) are indicated in the bottom-left corner. Red boxes highlight representative regions for visual comparison.

4 Experiments

4.1 Experimental Setup

Implementation Details. We implement HHC-GS based on Octree-GS (Ren et al., 2025) and conduct experiments on an Intel Core i9-14900K CPU with a single NVIDIA RTX 5080 GPU. We follow Octree-GS for hierarchical anchor initialization, growing, and pruning. After topology stabilization, we freeze the octree and use the optimized full model as a frozen teacher to jointly optimize the compact representation and Gaussian parameters. Other training and rendering settings follow Octree-GS, with method-specific hyperparameters provided in the supplementary material.

Datasets and Metrics. We evaluate our method on three large-scale datasets: MatrixCity (Li et al., 2023), BungeeNeRF (Xiangli et al., 2022), and UrbanScene3D (Lin et al., 2022), covering city-scale scenes with diverse spatial extents and viewing scales. We follow the official or commonly adopted train-test splits for all datasets. Reconstruction quality is evaluated using PSNR, SSIM, and LPIPS. Size denotes the total storage of the inference representation, including model parameters, hierarchical structures, auxiliary networks, and encoded data, while excluding training states and runtime buffers. We also report loaded map size, peak GPU memory, rendering FPS, and end-to-end latency.

Baselines. We compare HHC-GS with representative 3D Gaussian Splatting methods, including Scaffold-GS (Lu et al., 2024) and Octree-GS (Ren et al., 2025) for anchor-based and hierarchical LoD representations, HAC++ (Chen et al., 2025b), LightGaussian (Fan et al., 2024), and CAT-3DGS (Zhan et al., 2025) for compact and compressed representations, LapisGS (Shi et al., 2025), PCGS (Chen et al., 2026), and CLoD-GS (Cheng et al., 2025) for progressive and resource-adaptive representations. We use the official implementations and follow their recommended configurations whenever available.

4.2 Experimental Results And Analysis

Overall Performance. As shown in Tab. 1, HHC-GS achieves the smallest model size across all three datasets while preserving competitive reconstruction quality. It performs best on MatrixCity and UrbanScene3D and remains competitive on BungeeNeRF, where Octree-GS achieves slightly better reconstruction quality at a substantially larger storage cost. The qualitative results in Fig. 3 further show improved preservation of fine structures and local details, validating the effectiveness of ancestor-conditioned feature generation for compact high-quality scene representation.

Budget-Aware Reconstruction. We further evaluate reconstruction under increasing loaded-representation budgets and report the actual loaded size of each feasible hierarchy prefix in Fig. 4.

Table 1: Quantitative comparison on MatrixCity, BungeeNeRF, and UrbanScene3D. Best results are **bold**, second best are underlined.

Method	MatrixCity				BungeeNeRF				UrbanScene3D			
	PSNR ↑	SSIM ↑	LPIPS ↓	Size ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Size ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Size ↓
Scaffold-GS	20.59	0.695	0.396	397.1	26.62	0.865	0.241	183.0	22.06	0.793	0.225	183.5
Octree-GS	22.40	0.728	0.332	623.6	**28.10**	**0.919**	**0.093**	317.8	23.42	0.856	0.204	220.4
LightGaussian	20.44	0.707	0.423	219.8	24.52	0.825	0.255	87.3	22.13	0.791	0.219	82.1
HAC++	21.69	0.732	0.340	93.4	27.17	0.879	0.196	20.8	23.09	0.834	0.231	24.6
CAT-3DGS	20.75	0.683	0.339	115.0	27.35	0.886	0.183	26.6	24.38	0.862	0.176	52.3
LapisGS	21.48	0.712	0.346	509.8	23.62	0.775	0.263	884.1	23.20	0.855	0.194	263.8
CLoD-GS	22.39	0.725	0.253	167.4	27.09	0.885	0.150	89.6	23.94	0.844	0.185	53.0
PCGS	22.17	0.740	0.251	94.5	27.15	0.872	0.213	22.8	23.47	0.851	0.193	19.2
Ours	**22.78**	**0.765**	**0.240**	**82.5**	27.83	0.897	0.116	**20.4**	**24.73**	**0.882**	**0.165**	**15.3**

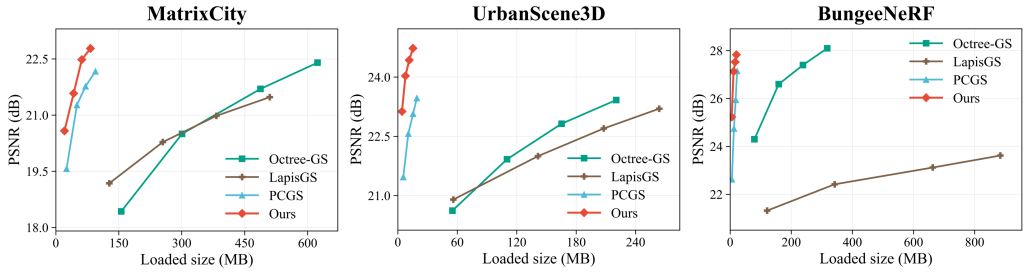


Figure 4: Resource-adaptive reconstruction on MatrixCity, UrbanScene3D, and BungeeNeRF. Each method is evaluated at four progressively larger representation operating points, and the actual amount of loaded representation is shown on the x-axis. All methods use the same dataset split and evaluation protocol.

Across all three datasets, HHC-GS maintains strong reconstruction quality with compact loaded representations, particularly under limited budgets. The results show that the hierarchical representation supports different loaded-representation budgets while preserving full-budget reconstruction quality.

Runtime and Memory Efficiency. As shown in Table 2, our method achieves a favorable balance between memory efficiency and computational cost. The compact hierarchical representation reduces the runtime memory footprint, while the additional ancestor-conditioned feature generation introduces only moderate overhead, while preserving resource-adaptive rendering and consistent local feature generation. The resulting runtime remains practical for large-scale rendering under memory constraints.

Table 2: Runtime efficiency comparison on BungeeNeRF. GPU Memory denotes peak inference memory usage, FPS denotes rendering-only throughput, and Latency denotes average end-to-end processing time per frame.

Method	GPU Memory (GB) ↓	FPS ↑	Latency (ms) ↓
Scaffold-GS	2.85	264	6.2
Octree-GS	3.53	136	10.6
LightGaussian	**1.71**	**294**	**4.5**
HAC++	1.96	216	5.9
CAT-3DGS	3.09	154	8.5
CLoD-GS	2.65	175	6.6
PCGS	2.39	248	6.1
Ours	1.82	205	6.3

Local Map Loading. Table 3 evaluates the consistency between local and full-map representations. HHC-GS requires less local data than the anchor-based baselines, while ancestor closure adds only limited loading overhead. Without closure, both reconstruction quality and feature consistency de-

Table 3: Local map loading on MatrixCity. Loaded Anchors and Loaded Data measure the local representation used for rendering. Closure Overhead is the extra cost of ancestor completion. PSNR Drop measures quality loss relative to the complete map, and Feature Error measures local-to-full feature discrepancy.

Method	Loaded Anchors(%) ↓	Loaded Data(MB) ↓	Closure Overhead(%) ↓	PSNR Drop(dB) ↓	Feature Error↓
Scaffold-GS	23.6	106.5	–	0.21	–
Octree-GS	11.5	89.4	–	0.18	–
Ours (w/o Closure)	9.6	14.1	0	0.63	7.5×10^{-2}
Ours	10.8	15.8	12.1	0.06	1.8×10^{-6}

Table 4: Ablation study of the proposed components on MatrixCity. Full PSNR evaluates complete-model reconstruction quality, while Avg. Prefix PSNR and LPIPS measure average quality over non-full prefixes.

Method	Full PSNR ↑	Avg. Prefix PSNR ↑	Avg. Prefix LPIPS ↓	Size (MB) ↓
Independent Feature	**23.11**	**21.86**	**0.263**	167.3
Low-dim Only	20.36	18.84	0.356	**81.6**
Parent Only	21.72	20.27	0.345	<u>82.3</u>
Euclidean Context	22.45	20.64	0.323	83.1
w/o Teacher	22.71	20.92	0.342	82.5
w/o Multi-prefix	22.58	20.37	0.327	82.6
Ours (full)	<u>22.78</u>	<u>21.13</u>	<u>0.276</u>	82.5

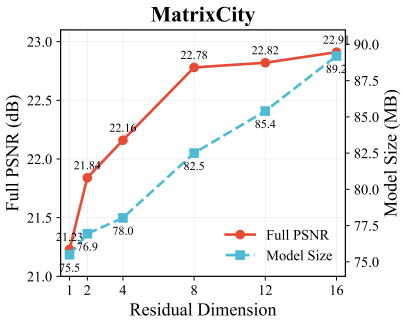


Figure 5: Effect of the residual dimension k on reconstruction quality and model size on MatrixCity.

grade. With ancestor closure, the feature discrepancy becomes negligible and the local rendering remains close to the full-map result. This confirms that ancestor-closed loading preserves the global feature-generation rule during local rendering.

4.3 Ablation Studies

Table 4 evaluates the contributions of compact feature representation, hierarchical context, and budget-aware training. Independent Feature directly stores a complete feature vector for every node without ancestor-conditioned prediction, achieving the highest reconstruction quality but requiring substantially larger storage. Low-dim Only significantly reduces storage but degrades reconstruction quality. Adding parent context improves performance, while full ancestor context provides further gains, showing the advantage of exploiting broader hierarchical dependencies. Replacing hyperbolic modeling with Euclidean context reduces both full-model and prefix quality at a comparable model size. Removing the teacher or multi-prefix supervision mainly degrades prefix quality, with multi-prefix supervision having a larger impact on intermediate representations.

We further study the residual dimension k in Fig. 5. Increasing k from 1 to 8 consistently improves reconstruction quality, while larger values provide only marginal gains with increased storage. We therefore set $k = 8$ as the default residual dimension.

5 Conclusion

In this paper, we presented HHC-GS, a compact multiscale Gaussian mapping framework for large-scale scene reconstruction under varying loaded-representation budgets. HHC-GS uses hierarchical ancestor context to generate non-root anchor features, requiring only low-dimensional node-specific variables to be stored. Experiments show that HHC-GS reduces representation storage while maintaining reconstruction quality across hierarchy prefixes and supporting compact local loading with consistent feature generation. Future work will focus on reducing context-generation overhead and extending HHC-GS to larger and more complex scenes.

AI USE STATEMENT

In this work, we used generative AI tools for reference organization, language polishing, grammar checking, and improving the wording of selected parts of the manuscript. We did not use generative AI tools to propose the core research methodology, generate experimental data, modify experimental results, or replace the authors' scientific judgment. We also did not use generative AI tools to generate unverified references.

We have reviewed and verified all AI-assisted content to ensure the accuracy of the technical descriptions, experimental results, mathematical notation, and citations. The research methodology, experimental design, implementation, result analysis, and final conclusions were developed, conducted, and validated by the authors.

We take responsibility for the final content of this work, including any text or other artifacts produced or revised with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To facilitate the reproducibility of this work, we provide a complete description of HHC-GS, including the training objective and experimental setup, in the main paper. Additional implementation details, hyperparameter settings, training configurations, quantitative results, and relevant data descriptions are provided in the appendix. All experiments are conducted on publicly available datasets using the official or commonly adopted data splits and baseline configurations. Upon acceptance, we will release the source code and experimental configurations to further support reproducibility.

REFERENCES

Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. Hyperbolic graph convolutional neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32, 2019.

Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Hac: Hash-grid assisted context for 3d gaussian splatting compression. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 422–438, Cham, 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.

Yihang Chen, Mengyao Li, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Pcgs: Progressive compression of 3d gaussian splatting. In *The 40th Annual AAAI Conference on Artificial Intelligence*, 2026.

Zhigang Cheng, Mingchao Sun, Yu Liu, Zengye Ge, Luyang Tang, Mu Xu, Yangyan Li, and Peng Pan. Clod-gs: Continuous level-of-detail via 3d gaussian splatting. In *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR)*, 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 A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 140138–140158, 2024.

Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of gaussians. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 165–181, Cham, 2024.

Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, et al. Hyperbolic attention networks. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

Zhilin Guo, Boqiao Zhang, Hakan Aktas, Kyle Fogarty, Jeffrey Hu, Nursena Koprucu Aslan, Wenzhao Li, Canberk Baykal, Albert Miao, Josef Bengtson, Chenliang Zhou, Weihao Xia, Cristina Nader Vasconcelos, and Cengiz Oztireli. Matryoshka gaussian splatting. *arXiv preprint arXiv:2603.19234*, 2026.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21537–21546, 2025.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4), July 2023.

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 Trans. Graph.*, 43(4), July 2024.

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

Yixuan Li, Lihan Jiang, Linning Xu, Yuanbo Xiangli, Zhenzhi Wang, Dahua Lin, and Bo Dai. Matrixcity: A large-scale city dataset for city-scale neural rendering and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3205–3215, October 2023.

Jiaqi Lin, Zhihao Li, Xiao Tang, Jianzhuang Liu, Shiyong Liu, Jiayue Liu, Yangdi Lu, Xiaofei Wu, Songcen Xu, Youliang Yan, and Wenming Yang. Vastgaussian: Vast 3d gaussians for large scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5166–5175, June 2024.

Liqiang Lin, Yilin Liu, Yue Hu, Xingguang Yan, Ke Xie, and Hui Huang. Capturing, reconstructing, and simulating: The urbanscene3d dataset. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 93–109, 2022.

Shiyong Liu, Xiao Tang, Zhihao Li, Yingfan He, Chongjie Ye, Jianzhuang Liu, Binxiao Huang, Shunbo Zhou, and Xiaofei Wu. Occlugaussian: Occlusion-aware gaussian splatting for large scene reconstruction and rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 26643–26652, October 2025a.

Yang Liu, Chuanchen Luo, Lue Fan, Naiyan Wang, Junran Peng, and Zhaoxiang Zhang. Citygaussian: Real-time high-quality large-scale scene rendering with gaussians. In *European Conference on Computer Vision*, pp. 265–282. Springer, 2025b.

Yang Liu, Chuanchen Luo, Zhongkai Mao, Junran Peng, and Zhaoxiang Zhang. Citygaussianv2: Efficient and geometrically accurate reconstruction for large-scale scenes. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR)*, volume 2025, pp. 84324–84341, 2025c.

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.

Wieland Morgenstern, Florian Barthel, Anna Hilsmann, and Peter Eisert. Compact 3d scene representation via self-organizing gaussian grids. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 18–34, 2025.

K. L. Navaneet, Kossar Pourahmadi Meibodi, Soroush Abbasi Koohpayegani, and Hamed Pirsiavash. Compgs: Smaller and faster gaussian splatting with vector quantization. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 330–349, Cham, 2025.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30, 2017.

Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the Lorentz model of hyperbolic geometry. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3779–3788. PMLR, 10–15 Jul 2018.

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

Frederic Sala, Chris De Sa, Albert Gu, and Christopher Re. Representation tradeoffs for hyperbolic embeddings. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4460–4469. PMLR, 10–15 Jul 2018.

Yunji Seo, Young Sun Choi, HyunSeung Son, and Youngjung Uh. Flod: Integrating flexible level of detail into 3d gaussian splatting for customizable rendering. *ACM Trans. Graph.*, 44(4), July 2025.

Yuang Shi, Simone Gasparini, Géraldine Morin, and Wei Tsang Ooi. LapisGS: Layered progressive 3D Gaussian splatting for adaptive streaming. In *International Conference on 3D Vision, 3DV 2025, Singapore, March 25-28, 2025*. IEEE, 2025.

Seungjoo Shin, Jaesik Park, and Sunghyun Cho. Locality-aware gaussian compression for fast and high-quality rendering. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR)*, volume 2025, pp. 63–90, 2025.

Zhiye Tang, Lingzhuo Liu, Shengjie Jiao, Qiudan Zhang, Junhui Hou, You Yang, and Xu Wang. Progs: Towards progressive coding for 3d gaussian splatting, 2026.

Peihao Wang, Yuehao Wang, Dilin Wang, Sreyas Mohan, Zhiwen Fan, Lemeng Wu, Ruisi Cai, Yu-Ying Yeh, Zhangyang Wang, Qiang Liu, and Rakesh Ranjan. Steepest descent density control for compact 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26663–26672, June 2025.

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 A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 51532–51551. Curran Associates, Inc., 2024.

Felix Windisch, Thomas Köhler, Lukas Radl, Mattia D'urso, Michael Steiner, Dieter Schmalstieg, and Markus Steinberger. A lod of gaussians: Out-of-core training and rendering for seamless ultra-large scene reconstruction. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, SIGGRAPH Conference Papers '26, 2026.

Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. Bungeenerf: Progressive neural radiance field for multi-scale scene rendering. In *Computer Vision – ECCV 2022*, pp. 106–122, 2022.

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

Zhensheng Yuan, Haozhi Huang, Zhen Xiong, Di Wang, and Guanghua Yang. Robust and efficient 3d gaussian splatting for urban scene reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 26209–26219, October 2025.

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 *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR) (ICLR)*, 2025.

Wei Zhang, Shengkai Yu, Shiqiang Gong, Qi Zhang, Qiang Li, and Qi Wang. Hico-gs: Hierarchical context aggregation and geometric consistency for octree gaussian splatting. In *Proceedings of the 34th ACM International Conference on Multimedia*, 2026.

Zihan Zheng, Zhenlong Wu, Houqiang Zhong, Yuan Tian, Ning Cao, Lan Xu, Jiangchao Yao, Xiaoyun Zhang, Qiang Hu, and Wenjun Zhang. 4dgcpro: Efficient hierarchical 4d gaussian compression for progressive volumetric video streaming. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 100860–100885, 2025. doi: 10.52202/085713-3376.

A Appendix

A.1 Additional Implementation Details

Training details. After the densification and pruning stage described in Sec. 4.1, we freeze the hierarchy topology and retain the optimized full representation as the teacher. The teacher parameters remain fixed throughout the subsequent compact training stage, and its full-level renderings for the training views are precomputed for the consistency loss in Eq. 12. The compact representation is then jointly optimized for $N_{\text{joint}} = 30{,}000$ iterations. Generated non-root features are evaluated recursively from coarse to fine levels, while anchors at the same level are processed in parallel. The remaining optimization and hierarchy-construction settings follow Octree-GS unless otherwise specified.

Dataset sequences. For MatrixCity (Li et al., 2023), we use the street-view portion of the *Small City* scene. For BungeeNeRF (Xiangli et al., 2022), we evaluate on *Amsterdam*, *Bilbao*, *Hollywood*, *Pompidou*, *Quebec*, and *Rome*, following the commonly used large-scale Gaussian splatting benchmark setting. For UrbanScene3D (Lin et al., 2022), we use the public *Residence* and *Sci-Art* sequences. We use the complete released sequences for BungeeNeRF and UrbanScene3D rather than spatially cropped subregions.

Method-specific settings. The method-specific hyperparameters used throughout the experiments are listed in Table 5.

Table 5: Method-specific hyperparameters used in HHC-GS.

Parameter	Description	Value
d_u	Structural embedding dimension	4
k	Residual code dimension	8
c	Curvature magnitude ($-c$)	1.0
τ	Ancestor aggregation temperature	0.5
ρ	Residual norm bound	0.5
m_r	Radial hierarchy margin	0.1
η	Hierarchical relation loss weight	0.01
γ	Teacher consistency loss weight	0.1
N_{joint}	Joint optimization iterations	30,000

A.2 Multi-prefix Training Details

Teacher filtering. The frozen teacher provides the full-level rendering $\bar{I}_L(v)$ for each training view. For prefix P_b, the same fixed level-specific operator G_b is applied to both the student rendering and the teacher rendering, as defined in Eq. 12. The filtering parameters are fixed before joint optimization and are not updated during training. The consistency term compares only the filtered outputs,

$$\mathcal{L}^b_{\text{con}}(v) = \frac{\left\| G_b[I_b(v)] - G_b[\bar{I}_L(v)] \right\|_1}{|\Omega|}. \tag{15}$$

This supervision does not assume that lower prefixes form image-space residuals of the complete representation. Instead, G_b restricts the consistency target to the image content retained by the corresponding level-specific filtering operator.

Prefix-wise Reconstruction Results Table 6 reports the reconstruction quality of individual prefixes for the training variants evaluated in Table 4. This separates the effect on intermediate prefixes from the complete representation, which is obscured when only the average prefix metric is reported.

Table 6: Prefix-wise reconstruction results on MatrixCity. P_3 denotes the complete hierarchy.

Method	P_0 PSNR	P_1 PSNR	P_2 PSNR	P_3 PSNR	Avg. Prefix PSNR
w/o Teacher	19.65	21.23	21.88	22.71	20.92
w/o Multi-prefix	18.63	20.52	21.96	22.58	20.37
Ours (full)	19.30	21.68	22.41	22.78	21.13

A.3 Local Map Loading Details

Loading statistics. Following the loading definition in Sec. 3.4, the *w/o Closure* variant uses the view-selected anchor set $S_b(v)$ directly without ancestor completion. The loaded-anchor ratio is computed as

$$R_{\text{anchor}} = \frac{|C_b(v)|}{N} \times 100\%, \tag{16}$$

where N is the number of anchors in the complete map. Loaded Data measures the serialized representation associated with the loading set and excludes generated feature caches and rendering buffers. We define the closure overhead with respect to the unclosed candidate set as

$$R_{\text{closure}} = \frac{S(C_b(v)) - S(S_b(v))}{S(S_b(v))} \times 100\%. \tag{17}$$

where $S(\cdot)$ denotes the amount of stored representation data associated with an anchor set.

Local-to-full consistency. Feature Error measures the discrepancy between features generated from the local loading set and those generated from the complete map for the same anchors:

$$E_{\text{feat}} = \frac{1}{|S_b(v)|} \sum_{i \in S_b(v)} \left\| f_i^{\text{local}} - f_i^{\text{full}} \right\|_2. \tag{18}$$

With fixed topology and model parameters, Eq. **??** provides all ancestors required by the recursive generation rule. Therefore,

$$f_i^{\text{local}} = f_i^{\text{full}}, \qquad i \in S_b(v). \tag{19}$$

and the theoretical feature error is zero up to numerical precision. Without ancestor closure, this equivalence is not guaranteed.

PSNR Drop is computed relative to the corresponding full-map rendering at the same hierarchy level:

$$\Delta_{\text{PSNR}} = \text{PSNR}\big(I_b^{\text{full}}, I^{\text{gt}}\big) - \text{PSNR}\big(I_b^{\text{local}}, I^{\text{gt}}\big). \tag{20}$$

A.4 Additional Runtime and Resource Details

Timing protocol. Following the evaluation convention of Octree-GS (**?**), we synchronize CUDA before and after each timed frame and exclude the first five views as warm-up. We distinguish steady-state GPU processing time from end-to-end latency. For prefix P_b, the former is defined as

$$t_b^{\mathrm{gpu}} = t_b^{\mathrm{sel}} + t_b^{\mathrm{feat}} + t_b^{\mathrm{dec}} + t_b^{\mathrm{rast}}, \tag{21}$$

where the four terms denote prefix and visibility selection, ancestor-conditioned feature generation, Gaussian decoding, and rasterization, respectively. The reported rendering throughput is

$$\mathrm{FPS}_b = \left(\frac{1}{N-5} \sum_{v=6}^{N} t_b^{\mathrm{gpu}}(v) \right)^{-1}. \tag{22}$$

End-to-end latency additionally includes construction of the ancestor-closed loading set and transfer of the required compact representation:

$$t_b = t_b^{\mathrm{load}} + t_b^{\mathrm{gpu}}. \tag{23}$$

The serialized representation is kept in host memory during evaluation; disk I/O and image saving are excluded from both measurements.

Memory and feature caching. Peak GPU memory is measured after warm-up by resetting the CUDA peak-memory statistics before each test view and recording the maximum allocated memory during loading, feature generation, and rendering. We report the maximum over all test views, consistent with the definition of M_b in Eq. (14). Generated features are cached only for anchors in the currently loaded ancestor-closed set; no persistent full-map feature cache is retained. This prevents feature caching from eliminating the memory advantage of budgeted loading.

B Limitations

HHC-GS is designed for static large-scale scenes with calibrated images and a fixed hierarchical anchor structure. Since compact optimization is performed after topology freezing, the current formulation is less suitable for dynamic scenes or online mapping where the hierarchy changes frequently. Its benefit also depends on the depth and quality of the hierarchy, and may be reduced when only limited ancestor context is available.

The compact representation reduces storage but introduces additional context-generation cost. Moreover, the current budget controls loaded representation size rather than peak GPU memory, which also depends on feature caches and rendering buffers. The block partition further introduces a trade-off between root-feature overhead and local loading efficiency, so a fixed block configuration may not be optimal for all scenes.