3D Gaussian Splatting Compression with Object Scalability

Ruixiang Xue, Tong Chen[†], and Zhan Ma

Nanjing University, Nanjing, China
xrxee@smail.nju.edu.cn, chentong@nju.edu.cn, mazhan@nju.edu.cn
[†]Corresponding author.

Abstract. We introduce a framework towards scalable, finer-grained object-level 3DGS compression. First, a post-training method named RecastGS is proposed to reorganize pretrained 3DGS into a layered representation and progressively distills cumulative submodels to improve rate–distortion efficiency. Leveraging multi-view SAM predictions from user click prompts, Gaussians are further partitioned into user-defined regions of interest (ROI), enabling region-adaptive quality control without retraining. Second, built upon this reorganized region-aware layered hierarchy, a feed-forward 3DGS compression method named LayeredCGS is proposed to compress position using a lightweight point cloud codec and attributes with a layer-wise context model to exploit cross-layer correlations. Extensive experiments show that LayeredCGS achieves 35% BD-Rate gain over the existing feed-forward method FCGS. With progressive distillation in RecastGS enabled, our method further outperforms most per-scene optimization methods. Moreover, the proposed method supports ROI-aware compression and flexible bitstream truncation, achieving up to 2 dB higher ROI PSNR at comparable bitrates compared with the uniform quality allocation baseline while enabling low-latency preview and progressive quality refinement. The code will be released at https://github.com/RuixiangXue/ScalableGSC.

Keywords: 3D Gaussian Splatting · Compression · Region of Interest

1 Introduction

3D Gaussian Splatting (3DGS) [30] has substantially advanced photorealistic 3D scene synthesis by enabling high-fidelity novel view synthesis with real-time rendering. These advantages make 3DGS a promising solution for diverse immersive applications. Unlike implicit Neural Radiance Fields (NeRF) [45], 3DGS explicitly represents scenes using millions of Gaussian primitives, resulting in prohibitive memory footprints (often exceeding 500 MB per scene) that severely limit practical deployment in resource-constrained scenarios.

To address this bottleneck, a wide range of 3DGS compression methods [2] have been proposed. Although these methods achieve strong compression performance, most of them focus on compressing the entire scene holistically. In interactive graphics applications, however, users typically engage with only a small

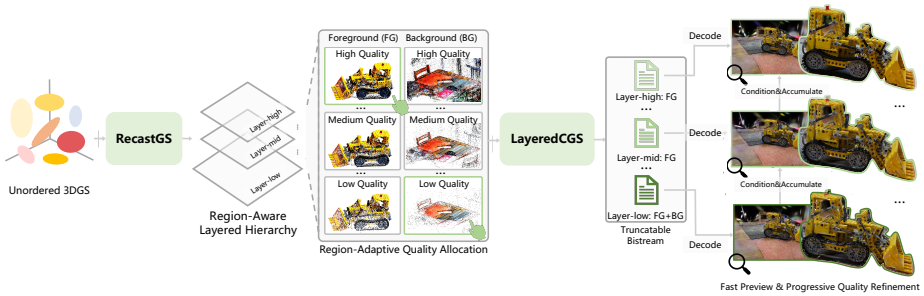


Fig. 1: Object-level scalable compression for 3DGS. Given an unordered 3DGS model, it is reorganized into a region-aware layered hierarchy for region-adaptive quality allocation using RecastGS. The layered feed-forward compressor LayeredCGS then generates a truncatable bitstream for fast preview and progressive quality refinement. Each quality level corresponds to a submodel of cumulative layers, and each layer is encoded as an independent bitstream segment.

portion of the scene (e.g., regions of interest, ROIs). This mismatch leads to inefficient resource usage, especially in large-scale scenes. While several works [6,12, 25,59] organize unordered Gaussian primitives into layered representations to enable progressive compression, they only support global quality control and lack finer-grained, object-level, or region-aware adaptation. We address this problem with two sequential rather than alternative components: **representation reorganization** (RecastGS) and **layered feed-forward compression** (LayeredCGS), as illustrated in Fig. 1.

The vanilla 3DGS lacks intrinsic hierarchy and object-level organization, making region-adaptive quality control difficult. To address this structural limitation, we propose RecastGS, which reorganizes unordered Gaussian primitives into a region-aware layered hierarchy. Starting from a pretrained 3DGS model, RecastGS first constructs a layered representation via overcomplete layer search and refines cumulative submodels through progressive distillation. In parallel, a segmentation module extracts object-level 3D masks from sparse user click prompts. By combining the layered representation with the extracted object-level masks, RecastGS establishes a recomposable representation that enables adaptive quality allocation between foreground and background regions under varying user preferences and bandwidth constraints.

Many compression methods [8, 17, 38, 61] tightly couple compression with the reconstruction process, requiring full access to the training pipeline. This limits their applicability to arbitrary pretrained models. In contrast, several works [14, 39, 51] explore post-training compression that operates directly on pretrained models. Among them, feed-forward methods [7] enable single-pass encoding of arbitrary pretrained 3DGS models without per-scene optimization, substantially reducing encoding latency to seconds. Nevertheless, existing feed-forward approaches still compress 3DGS holistically without supporting finer-grained region-aware adaptation. Building upon the reorganized representation

from RecastGS, we propose LayeredCGS, a layered feed-forward 3DGS compression method that generates truncatable bitstreams for low-latency preview and progressive quality refinement. We introduce a layer-wise context model that exploits decoded lower-layer Gaussians to model cross-layer dependencies. We also integrate RENO-GS, a lightweight point cloud geometry codec, for efficient position compression. Generally, this work makes the following contributions:

- We propose RecastGS, which reorganizes pretrained 3DGS into a region-aware layered hierarchy via overcomplete layer search, progressive distillation, and prompt-driven object extraction, enabling recomposable representations for region-adaptive quality control.
- We introduce LayeredCGS, a feed-forward compression method for layered 3DGS representations that exploits cross-layer dependencies through layer-wise context modeling and generates truncatable bitstreams for efficient progressive compression.
- Extensive experiments demonstrate improved rate–distortion performance over both feed-forward and per-scene optimized methods, while enabling ROI-aware progressive compression.

2 Related Work

2.1 3DGS Compression

Per-scene Optimized Compression Approaches achieve strong compression performance through scene-specific optimization. A major branch [8,9,17,35, 38,46,60,61] tightly couples reconstruction and compression into a joint optimization pipeline, improving rate–distortion performance via scene-specific pruning, quantization, or compact parameterizations (e.g., Scaffold-GS [42]). However, they require full access to the reconstruction, limiting their applicability to arbitrary pretrained models. Another branch [14, 34, 39, 48, 49, 51, 54, 64] performs compression as a post-training step, decoupling it from the original reconstruction process but still requiring scene-specific optimization. For example, LightGaussian [14] employs significance-based pruning with SH distillation and adaptive vector quantization. These methods still incur non-negligible encoding overhead due to the scene-specific optimization.

Feed-forward Compression Approaches [7, 41, 57, 66, 72] aims to compress arbitrary pretrained 3DGS models in a single forward pass without scene-specific optimization. A straightforward strategy is to treat 3DGS as point cloud-like data and adopt standard point cloud codecs [25, 66]. For example, HGSC [25] employs MPEG G-PCC (Octree) [1, 56] for geometry coding and RAHT [53] for attribute coding. FCGS [7] proposes a learning-based framework, compressing the attribute using a gaussian mixture probabilistic model for entropy coding. These approaches significantly reduce encoding latency and improve applicability due to their paradigm-level efficiency. However, existing designs typically

compress scenes holistically, making it difficult to support adaptive quality allocation for user-specified regions. Our proposed LayeredCGS belongs to the feed-forward approach and tackles this limitation.

2.2 Layered Representation in 3DGS Compression

Progressive compression improves user experience through fast preview and adaptive quality enhancement, which typically relies on layered representations. Several recent methods [6, 12, 25, 59] explore progressive compression for 3DGS. GoDe [12] builds a gradient-based Gaussian hierarchy, and RAVE [59] further re-estimates gradient between layers for continuous rate control. HGSC [25] constructs hierarchical anchor primitives via K-D tree partitioning and farthest point sampling. PCGS [6] introduces a progressive learnable mask to partition Gaussian primitives. These methods substantially improve the practicality of 3DGS for streaming but still operate at a global quality level, leaving room for finer-grained region-adaptive quality allocation.

Meanwhile, **Region-of-Interest (ROI) compression** has been widely studied in image, video, and point cloud coding to allocate higher quality to important regions under bitrate constraints [5,13,18,22,26,28,29,31,36,44,55,63,65,67]. Recently, several 3DGS segmentation methods [10, 20, 23, 27, 43, 52, 55, 68, 69, 74–76] have emerged, enabling the extraction of object-level masks from user interactions. Leveraging this capability for interactive region definition, we further couple it with a layered representation, enabling finer-grained region-adaptive quality allocation for ROI-aware progressive compression.

3 Proposed Method

Our method consists of two sequential components: RecastGS (Sec. 3.2 and Fig. 2) and LayeredCGS (Sec. 3.3 and Fig. 3). RecastGS first provides a region-aware layered representation, and LayeredCGS then compresses this representation into independently decodable bitstream segments. Before detailing these components, we briefly revisit the 3DGS Format and Layered Representation in 3DGS.

3.1 Preliminary

3DGS Format 3D Gaussian Splatting [30] represents a 3D scene using a set of Gaussian primitives, which can be parameterized as the position $\mu \in \mathbb{R}^{N_g \times 3}$, covariance matrix $\sum \in \mathbb{R}^{N_g \times 3 \times 3}$, opacity $\sigma \in \mathbb{R}^{N_g \times 1}$, and view-dependent RGB color $c \in \mathbb{R}^{N_g \times 3 \times (sh+1)^2}$. Here N_g denotes the number of Gaussian primitives, and sh is the spherical harmonics degree. Covariance matrix Σ can be further parameterized as rotation $r \in \mathbb{R}^{N_g \times 4}$ and scaling $s \in \mathbb{R}^{N_g \times 3}$. For rendering, a tile-based rasterizer sorts them in a front-to-back depth order and applies alpha blending after projecting them onto the image plane as 2D Gaussians.

Such a 3DGS model consists of an unordered set of Gaussian primitives. Several recent works [6,12,25,59] explore layered representations for 3DGS compression. Here, we briefly revisit the layered representation proposed in GoDe [12], which serves as the basis for our subsequent modeling.

Layered Representation In this layered representation, the number of Gaussians allocated to each layer follows an exponential schedule. Specifically, given the total number of layers L, the total number of Gaussians N_g, and the number of Gaussians in the first layer n_1, the cumulative number of Gaussians up to layer i is defined as

$$C_i = \left\lfloor n_1 \cdot \exp\left(\frac{\log N_g - \log n_1}{L-1} \cdot i \right) \right\rfloor, \quad i = 1, \ldots, L. \tag{1}$$

Then, the number of Gaussians assigned to each layer $\{n_2, n_3, \ldots, n_L\}$ is

$$n_i = C_i - C_{i-1}, \quad i = 2, \ldots, L, \tag{2}$$

Starting from layer L to 1, the importance score for each Gaussian primitive is iteratively computed within the remaining Gaussian set R_i based on the training view set V. At each layer, the n_i Gaussians with the lowest accumulated importance scores are selected and excluded from subsequent iterations, assigning less important primitives to higher layers. For each Gaussian $g \in R_i$, the cumulative importance score is defined as

$$s_g = \sum_{v \in V} \sum_{p \in \mathrm{P}(g)} \left\| \frac{\partial \mathcal{L}(v)}{\partial p} \right\|_2, \tag{3}$$

where $\mathrm{P}(g) = \{\boldsymbol{\mu}_g, \boldsymbol{\sigma}_g, \boldsymbol{c}_g, r_g, \boldsymbol{s}_g\}$ denotes the parameter set of g, and $\| \cdot \|_2$ denotes the ℓ_2 norm measuring the magnitude of parameter-wise gradients. The loss function follows vanilla 3DGS.

3.2 RecastGS: From Unordered to Region-Aware Layered Hierarchy

As illustrated in Fig. 2, RecastGS consists of two main components: Layered Representation Construction and Object-Level Segmentation. The former includes overcomplete layer search and progressive distillation, while the latter is achieved via prompt-driven object extraction.

Overcomplete Layer Search We study the impact of hyperparameters involved in the layered representation of Sec. 3.1 and observe that the total layer number L plays a critical role in the quality of the selected Gaussian subsets. Increasing L leads to finer-grained partitions of the Gaussian set, allowing the gradient-based importance score to better differentiate primitives with varying contributions to reconstruction fidelity. However, increasing L leads to a higher computational cost of subsequent finetuning to achieve the same fidelity as a

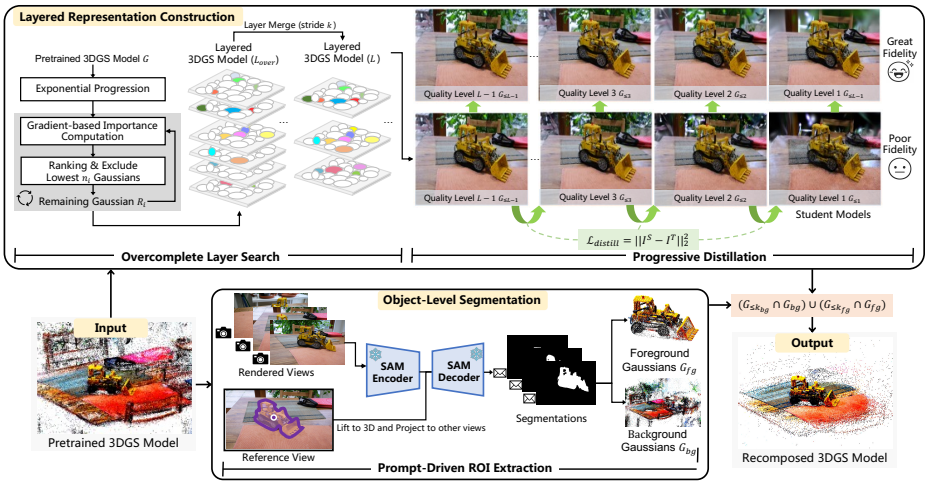


Fig. 2: Overview of RecastGS. Given a pretrained 3DGS model, RecastGS reorganizes unordered Gaussian primitives into a region-aware layered hierarchy. The upper branch constructs a layered representation via overcomplete layer search and progressive distillation, while the lower branch extracts object-level masks from user prompts through prompt-driven object extraction. The output of both branches is recomposed to enable region-adaptive quality allocation.

smaller L. To balance selection quality and computational efficiency, we adopt a simple yet effective overcomplete layer search strategy. We first construct an Overcomplete Layered 3DGS Model with an expanded layer depth $L_{\text{over}} > L$ to obtain a fine-grained partition of Gaussian primitives. Then a uniform layer merge is employed to these layers with a fixed stride k, resulting in a reduced layer set of size $L = \lfloor L_{\text{over}}/k \rfloor$. Specifically, every k adjacent overcomplete layers are merged into one target layer, e.g., $(1, 2) \to 1$ when $k = 2$. As shown in Sec. 4.3, the proposed strategy achieves a superior rate–distortion trade-off compared to the naive configuration in GoDe. This modification only incurs a small computational overhead than GoDe (e.g., an average of 8.3 s in the Mip-NeRF360 dataset when $L_{over} = 16, L = 8$).

Progressive Distillation Although the overcomplete layer search yields a well-structured layered representation, the parameters in the pretrained 3DGS model are not optimized for this partitioning. We further conduct finetuning following GoDe [12]. A direct finetuning strategy as adopted in GoDe randomly samples a quality level $i \in \{1, \ldots, L\}$ at each iteration and optimizes only the Gaussian primitives assigned to layers $\leq i$ under full ground-truth image supervision.

However, such a finetuning strategy exhibits several limitations. First, it ignores the varying capacities of Gaussian subsets across quality levels. Subsets with fewer primitives have limited expressive power, and directly finetuning them with full ground-truth signals may cause unstable high-frequency fitting, harming optimization stability and reconstruction fidelity. Second, random quality

level sampling leads to unbalanced layer optimization, especially when L is large. Third, supervising with full ground-truth images incurs additional storage and data-access overhead, which is limited in practice.

To address these limitations, we propose a progressive distillation strategy for layered 3DGS representations. Let the input pretrained 3DGS model serve as the teacher model, and I^{T} denote the rendered image from it. At distillation stage i, we activate only the Gaussian primitives assigned to layers $\leq i$, forming a student submodel $G_{\leq i}$. The rendered output of the student model is denoted as I_i^{S}. The distillation is minimized:

$$\mathcal{L}_{\mathrm{distill}}^{i} = \left\| I_i^{\mathrm{S}} - I^{\mathrm{T}} \right\|_2^2 \tag{4}$$

We adopt a progressive distillation schedule, where the stage index i is sequentially decreased from $L-1$ to 1. At each stage, the submodel $G_{\leq i}$ is initialized from the optimized parameters of $G_{\leq (i+1)}$. This fine-to-coarse schedule starts from submodels closer to the full teacher and progressively removes less critical primitives, making each stage a local refinement rather than learning a low-capacity model from scratch. This results in a sequence of distilled submodels $\{G_{\leq L-1}, \ldots, G_{\leq 1}\}$. The full model $G_{\leq L}$ corresponds to the pretrained 3DGS. Experiments in Sec. 4.3 demonstrate that the proposed progressive distillation strategy achieves a better rate–distortion trade-off than the finetuning scheme of GoDe as well as a variant that distills stages from 1 to $L-1$.

Prompt-Driven Object Extraction Region-of-Interest (ROI)-aware compression optimizes bit allocation by prioritizing ROIs, which is particularly important for 3DGS compression because its large data volume often makes full-scene loading and decoding impractical. However, existing methods [6, 12, 25] only support the global quality control without finer-grained region-adaptive quality control. Meanwhile, as diverse user preferences exist, ROI-aware compression needs to be customizable. The object-level segmentation in RecastGS is optional and can be replaced by other 3DGS segmentation methods.

In this work, we adopt a 2D prompt-driven object extraction method [24] that maps user clicks to subsets of 3D Gaussians, enabling object-level segmentation without additional training. Given a 3DGS model $G = \{g_i\}_{i=1}^{N_g}$ with positions μ_i, the user provides a 2D point prompt p^{2D} on a reference rendered view. The 2D prompt p^{2D} is lifted to a 3D prompt p^{3D} by selecting a Gaussian whose projected center lies within a small neighborhood of p^{2D} and has positive depth:

$$p^{3D} = \arg\min_{\mu_i} \left\{ d(\mu_i) \;\middle|\; \|P\mu_i - p^{2D}\|_1 < \epsilon,\ d(\mu_i) > 0 \right\} \tag{5}$$

where P denotes the projection matrix of the reference view, $d(\mu_i)$ denotes depth, and ϵ defines the search radius. The 3D prompt p^{3D} then is projected to other rendered views to generate view-specific 2D prompts, which are used by SAM [32] to obtain foreground masks m_j. For each view j, a Gaussian g_i is labeled as

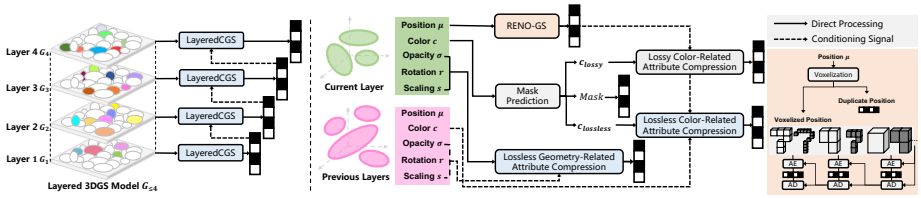


Fig. 3: Overview of LayeredCGS. Given a layered 3DGS representation, LayeredCGS compresses Gaussian primitives layer by layer, where each layer is compressed conditioned on previously decoded layers. Position is compressed using the lightweight RENO-GS point cloud geometry codec, while geometry- and color-related attributes are compressed using entropy models conditioned on previous layers.

foreground if its projected center falls inside the mask:

$$L_{ij} = \begin{cases} 1, & P_j \mu_i \in m_j, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Multi-view voting determines the final Gaussian membership. For each Gaussian g_i, we compute the foreground score:

$$s_i = \frac{1}{N} \sum_{j=1}^{N} L_{ij}, \tag{7}$$

where N denotes the number of views. The resulting subsets $G_{\text{fg}} = \{g_i \mid s_i > \tau\}$ and $G_{\text{bg}} = \{g_i \mid s_i \leq \tau\}$ define the foreground and background object segmentation. The τ denotes the threshold value

Recomposition Given a layered 3DGS model with cumulative quality levels $\{G_{\leq 1}, \ldots, G_{\leq L}\}$ from Layered Representation Construction and user-defined subsets $\{G_{\text{fg}}, G_{\text{bg}}\}$ from Object-Level Segmentation. We enable region-adaptive quality control through recomposition. Let k_{fg} and k_{bg} denote the target quality levels for the foreground and background regions, respectively. The recomposed Gaussian set is defined as

$$G^{\text{rec}} = \left(G_{\leq k_{\text{fg}}} \cap G_{\text{fg}}\right) \cup \left(G_{\leq k_{\text{bg}}} \cap G_{\text{bg}}\right). \tag{8}$$

Here, the $\cup$ and $\cap$ denote set union and intersection, respectively.

3.3 LayeredCGS: Layered feed-forward Compression of 3DGS

As illustrated in the left part of Fig. 3, we take $G_{\leq 4}$ as an example, assuming $k_{fg} = k_{bg} = 4$, that it can be decomposed into distinct layers $\{G_1, G_2, G_3, G_4\}$. Each layer corresponds to a disjoint subset of Gaussian primitives. To efficiently compress this layered structure, we introduce a layered feed-forward compression

method termed LayeredCGS. Compression proceeds layer by layer: G_1 is encoded first, and each subsequent layer G_i is compressed conditioned on the previously decoded layers $\{G_1, \ldots, G_{i-1}\}$.

As shown in the right part of Fig. 3 and described in Sec. 3.1, The Gaussian primitives are categorized into three groups: position μ, geometry-related attributes $\{\sigma, r, s\}$, and color-related attribute c.

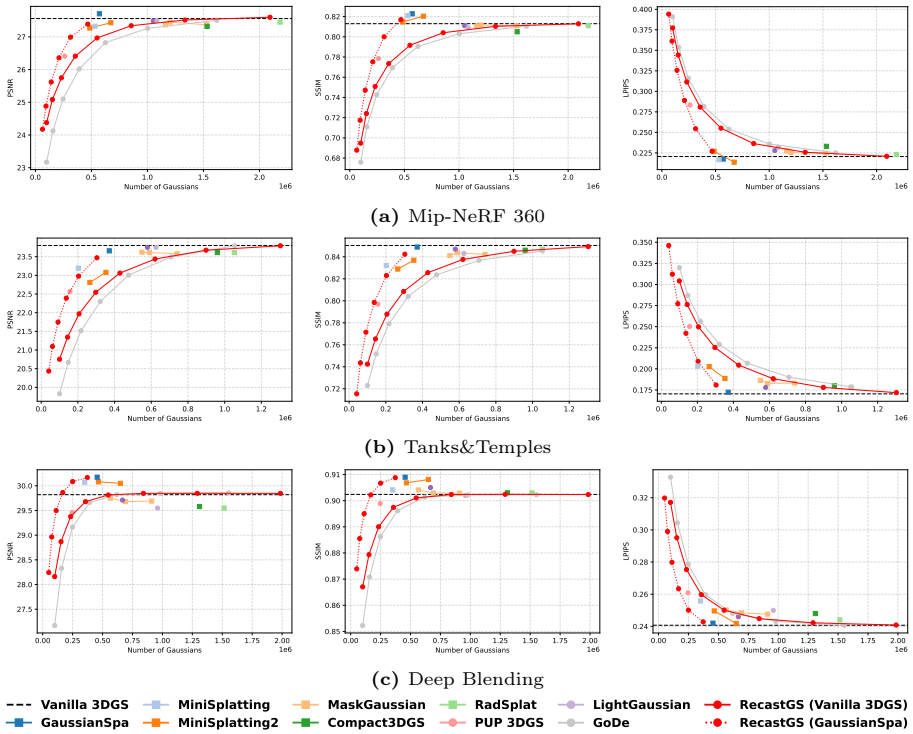
For the **position** component, most existing methods, e.g., FCGS [7], adopt G-PCC, which introduces non-trivial encoding latency. To improve efficiency, we employ RENO [70], a lightweight learning-based point cloud geometry codec, termed RENO-GS. Unlike G-PCC, RENO-GS performs one-shot sparse occupancy encoding and leverages the TorchSparse [58] engine for acceleration. Since RENO-GS operates on voxelized coordinates, and voxelization may produce a small fraction of duplicate positions. These duplicates are compressed losslessly with the DEFLATE algorithm [11].

Due to the high sensitivity of the **geometry-related attribute**, it is quantized and then losslessly compressed. For the **color-related attribute**, it is guided to a mask generated by a mask prediction network, splitting it into two groups, i.e., c_{lossy} and $c_{lossless}$. Such a mask is also compressed into the bitstream. Similar to the geometry-related attribute, $c_{lossless}$ is quantized and conducts lossless compression. c_{lossy} is compressed using an MLP-based encoder and decoder architecture. The compression of c_{lossy} and mask prediction follows the design of FCGS and is marked as gray in Fig. 3. Different from FCGS, LayeredCGS conditions the compression of G_i on previously decoded layers $\{G_1, \ldots, G_{i-1}\}$. They serve as contextual inputs to a layer-wise entropy model for both lossless geometry- and color-related attribute compression modules, effectively reducing inter-layer redundancy and improving coding efficiency.

The layer-wise context model consists of three branches: intra-Gaussian channel context, hyperprior, and inter-Gaussian spatial context. Their outputs are fused to parameterize a Gaussian Mixture Model (GMM) for entropy coding. The former two are inherited from FCGS. For inter-Gaussian context modeling, we construct a multi-resolution feature grid [47] from previously decoded Gaussians across the preceding layers. The grid comprises one 3D grid and three orthogonal 2D plane grids (xy, xz, yz). Decoded Gaussians splat their latent features onto the grids, while undecoded Gaussians query spatial context via trilinear interpolation. For a grid entry v, its context feature is computed by aggregating decoded latent features from previous layers:

$$f_v = \frac{\sum_{k \in \mathcal{N}(v)} w_k \hat{y}_k}{\sum_{k \in \mathcal{N}(v)} w_k}, \tag{9}$$

where $\hat{y}_k$ denotes the decoded latent feature of a Gaussian from previous layers, $\mathcal{N}(v)$ is the set of decoded Gaussians contributing to v, and w_k is a distance-based interpolation weight. The interpolated features are concatenated with a positional frequency encoding of the Gaussian center and processed by a context analyzer to predict the mean, scale, and mixture weights of the entropy model. These parameters are fused with intra-Gaussian channel context and hyperprior



(a) Mip-NeRF 360

(b) Tanks&Temples

(c) Deep Blending

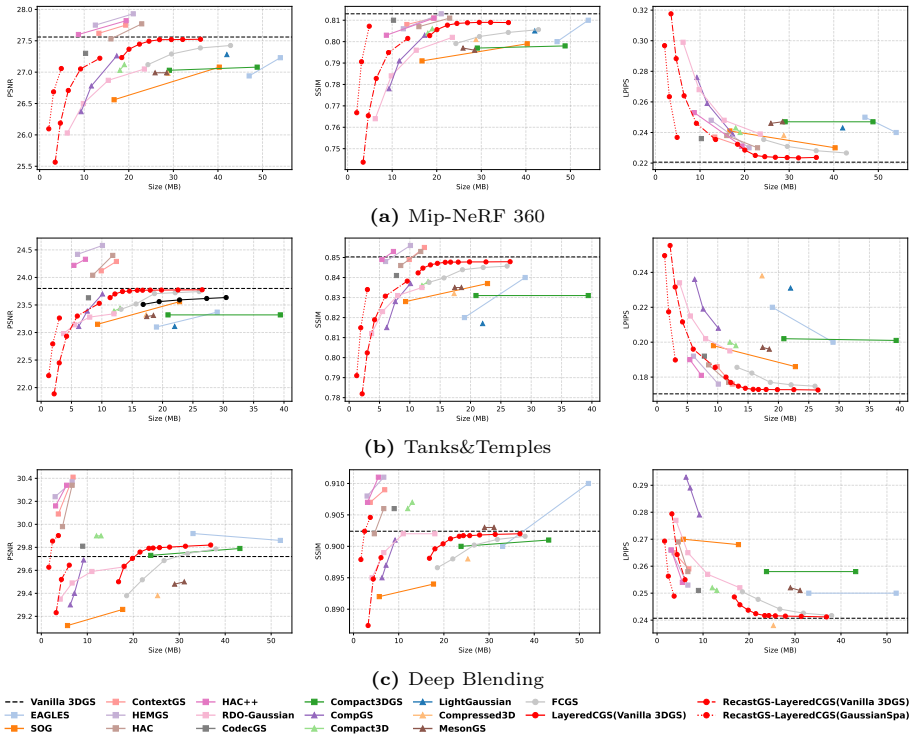
Fig. 4: Compaction performance comparison across Mip-NeRF 360, Tanks&Temples, and Deep Blending datasets. Detailed data can be found in the supplementary material.

signals to form a three-component GMM for entropy coding. More details about the network structure can be found in the supplementary material.

Optimization The lossless color- and geometry-related attribute compression modules, as well as RENO-GS, are retrained. For the lossless attribute compression modules, we optimize the rate loss, and the distortion item is excluded because the quantization step is pre-defined:

$$\mathcal{L} = \frac{bits_{total}}{N_g \times D}, \tag{10}$$

where N_g is the number of Gaussians and D denotes the attribute dimension (56 by default). $bits_{total} = bits_{geo} + bits_{color}$ represents the total bit consumption of geometry- and color-related attributes. During training, a layer is randomly sampled at each iteration, while all layers share the same network parameters. RENO-GS is trained separately using a cross-entropy objective for occupancy prediction. Other compression modules use pretrained weights from FCGS.



(a) Mip-NeRF 360

(b) Tanks&Temples

(c) Deep Blending

Fig. 5: Compression performance comparison across Mip-NeRF 360, Tanks&Temples, and Deep Blending datasets. Detailed data can be found in the supplementary material.

4 Experiments & Analysis

4.1 Experimental Setup

Datasets We train LayeredCGS on DL3DV-10K [37] dataset following FCGS [7], which contains around 7K real-world scenes. The pretrained 3DGS models are generated using the corresponding 960P images and SfM data, following the official implementation of vanilla 3DGS [30]. For testing, we select commonly used real-world scenes from MipNeRF360 [3], Tanks&Temples [33], and DeepBlending [21]. The evaluation adheres to the testing convention in 3DGS [30]. Per-view segmentation masks used for evaluating ROI-aware compression are obtained using SAM [32], and details are provided in the supplementary material.

Metrics The bitrate is computed based on the compressed file size or the number of retained gaussians, while fidelity is evaluated using PSNR, SSIM [62], and LPIPS [71]. To assess overall compaction/compression performance, we also adopt the BD-BR metric [4]. In addition, visualization of rendering is also vital to demonstrate the efficiency of our work. For ROI-aware compression, additional ROI-PSNR is included for evaluation.

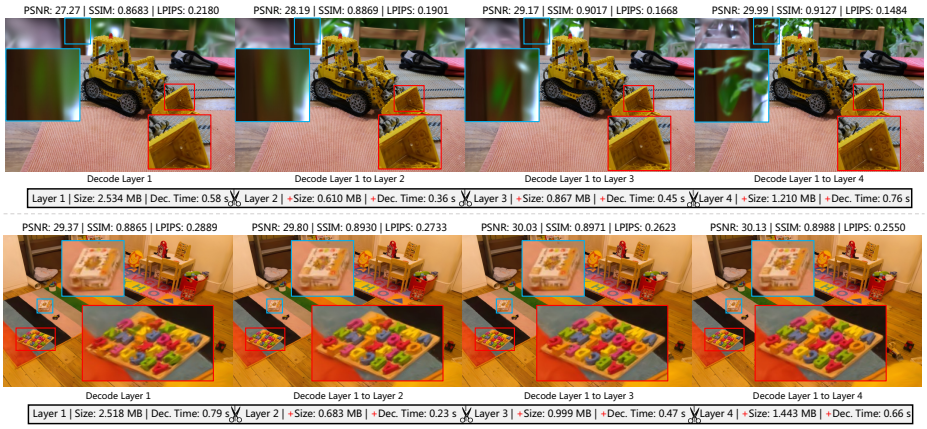
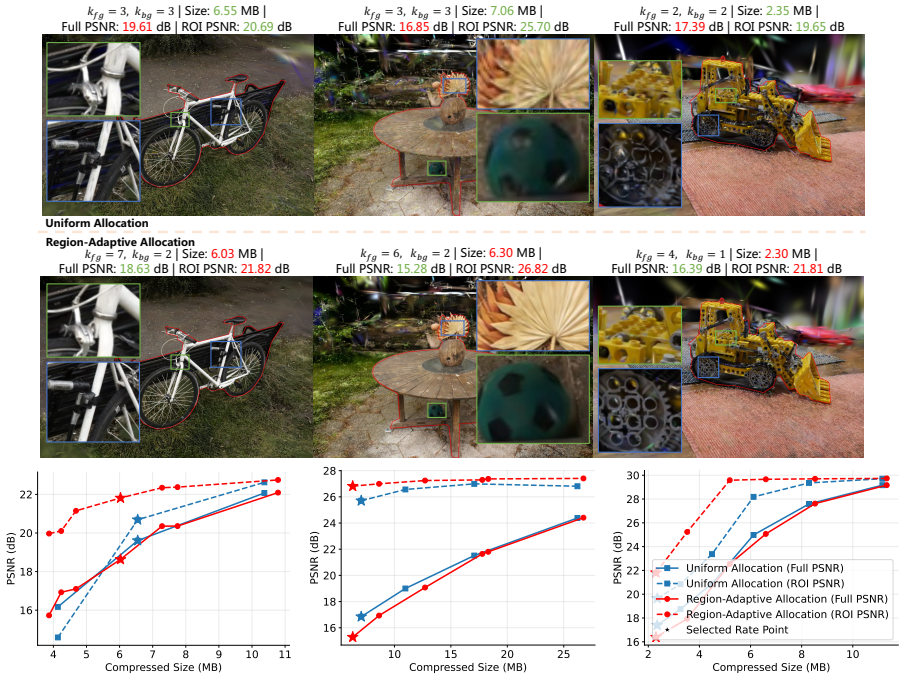


Fig. 6: Illustration of flexible bitstream truncation on *kitchen* and *playroom*.

Baselines As LayeredCGS belongs to feed-forward compression, we compare it with publicly available feed-forward methods, including FCGS [7]. Meanwhile, our framework also includes an optimization stage in RecastGS, namely progressive distillation. We also consider representative per-scene optimized methods. Post-training approaches include LightGaussian [14], Compressed3D [49], CompGS [39], Compact3DGS [34], Compact3D [48], and MesonGS [64]. Coupled optimization methods include HEMGS [38], HAC/HAC++ [8, 9], ContextGS [61], RDO-Gaussian [60], CodecGS [35], SOG [46], and EAGLES [17]. It is noted that ContextGS, HAC, HAC++, and HEMGS are based on anchor-based 3DGS [42], which is different from the data format used in our method.

To evaluate RecastGS alone, we further compare it with recent 3DGS pruning approaches. Among them, PUP 3DGS [19], LightGaussian [14], and GoDe [12] are post-training pruning methods, whereas GaussianSpa [73], MiniSplatting [15], MiniSplatting2 [16], MaskGaussian [40], Compact3DGS [34], and RadSplat [50] performs pruning in an integrated manner during 3DGS reconstruction.

Implementation Details In the overcomplete layer search of RecastGS, n_1, L_{over}, and L are set to 1×10^5, 16, and 8, respectively. Progressive distillation is performed for 30,000 iterations. In LayeredCGS, Gaussian positions μ are pre-quantized to 16 bits. The model is trained for 50 epochs using the Adam optimizer, with the learning rate linearly decayed from 2×10^{-4}. All experiments are conducted on an NVIDIA GeForce RTX 4090 GPU using PyTorch. The grid resolutions in layer-wise context model are $\{60, 80, 100\}$ for the 3D grid and $\{300, 400, 500\}$ for the 2D grids. The search radius ϵ and threshold τ in prompt-driven object extraction are 6 and 0.7, respectively.



(a) bicycle **(b)** garden **(c)** kitchen

Fig. 7: ROI-aware Compression Performance Comparison. Full image PSNR and ROI PSNR are both evaluated. The red boundaries indicate the ROI regions.

4.2 Experimental Evaluation

Compaction Performance Compared with GoDe, RecastGS achieves average BD-BR reductions of -24.90%/-18.69%/-12.90% across the MipNeRF 360, Tanks&Temples, and DeepBlending datasets for PSNR, SSIM, and LPIPS metrics, respectively. Here, RecastGS excludes prompt-driven object extraction, and both methods are applied to identical pretrained vanilla 3DGS models for fairness. The corresponding R-D curves are provided in Fig. 4. Optionally, the pretrained vanilla 3DGS models can be replaced with other sparsified variants, e.g., GaussianSpa, achieving the best rate-distortion performance. The average BD-BR reductions over GoDe reach -70.98%/-68.93%/-61.58%, indicating markedly enhanced compaction efficiency.

Compression Performance Compared with the feed-forward baseline FCGS, LayeredCGS achieves average BD-BR reductions of -34.91% / -38.42% / -34.83% on MipNeRF360, Tanks&Temples, and DeepBlending for PSNR, SSIM, and LPIPS metrics, respectively. Both methods compress the same pretrained vanilla 3DGS models without progressive distillation and prompt-driven object extraction in RecastGS to ensure a fair comparison. The only difference is that Layered-

CGS leverages the layered structure obtained from the overcomplete layer search. The detailed rate–distortion curves are shown in Fig. 5. When the pretrained 3DGS models are further refined using progressive distillation, i.e., RecastGS-LayeredCGS (Vanilla 3DGS), the bitrate is greatly reduced. Furthermore, when replacing vanilla 3DGS with sparsified reconstruction backbones such as GaussianSpa, i.e., RecastGS-LayeredCGS (GaussianSpa), the bitrate is reduced even further, achieving the lowest bitrate among all compared approaches. Overall, these results demonstrate the superior compression performance of our method. The compression/compaction results of other sparsified variants are detailed in the supplementary material.

Bitstream Truncation In the above analysis, we assessed overall rate-distortion efficiency. However, in streaming applications, latency plays a crucial role in the quality of experience. As shown in Fig. 6, the layered representation and progressive coding in our framework enable flexible bitstream truncation, adapting to varying resource constraints while supporting low-latency preview after decoding the initial level. Within around 0.7 seconds, our method provides a full-scene 3DGS preview. Subsequent progressive decoding incrementally enhances visual fidelity with minimal additional latency and bandwidth overhead. More results can be found in the supplementary material.

ROI-aware Compression We evaluate ROI-aware compression on the *bicycle*, *garden*, and *kitchen* scenes from Mip-NeRF360. As shown in Fig. 7, region-adaptive quality allocation consistently improves ROI PSNR at comparable compressed sizes compared with uniform quality allocation. Specifically, it achieves BD-BR reductions of -28.57% / -51.96% / -33.75% for ROI PSNR and degradation of 4.59% / 12.05% / 7.06% for full-image PSNR, respectively. This behavior reflects the intended bitrate redistribution toward regions of interest. For example, on *kitchen* at approximately 2.3 MB, ROI PSNR increases from 19.65 dB under uniform allocation to 21.81 dB (+2.16 dB) with region-adaptive allocation, significantly enhancing the foreground quality while slightly degrading the full-image quality. These results confirm that the proposed method enables effective object-level scalability. More results can be found in supplementary material.

4.3 Ablation Study

Module Analysis of RecastGS The contribution of each component in RecastGS is evaluated in Fig. 8(a). Using GoDe [12] as the baseline, we progressively integrate the proposed modules and report the corresponding BD-BR reductions. RecastGS achieves 6.06% BD-BR reduction compared to GoDe without finetuning, demonstrating that overcomplete layer search provides a better initialized layered representation. After enabling finetuning, we compare two scheduling strategies of progressive distillation in RecastGS. The bottom-up order achieves a 21.22% BD-BR reduction over the finetuned baseline, while the top-down order further improves it to 31.27%. The top-down strategy prioritizes high-frequency

components in early layers, leading to improved rate–distortion performance. It is noted that the overcomplete layer search is a one-shot preprocessing step that incurs only a small computational overhead (e.g., an average of 17.4s on the MipNeRF360 dataset when $L_{over} = 16, L = 8$), whereas progressive distillation requires around 15 minutes on average.

Module Analysis of LayeredCGS The contribution of each component in LayeredCGS is evaluated in Fig. 8(b). Using FCGS [7] as the baseline, we found that excluding RENO-GS and layer-wise context model still results in a 30.07% BD-BR reduction, indicating that the layered 3DGS representation from overcomplete layer search organizes Gaussian primitives into a more structured form that is favorable for compression. Introducing the layer-wise context model further achieves an additional 8.06% BD-BR reduction, reflecting the strong cross-layer correlations. RENO-GS contributes an additional 1.25% reduction and reduces decoding latency by around 1.7× due to its lightweight design.

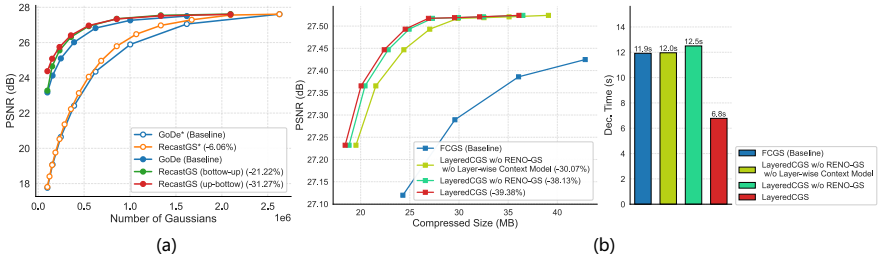


Fig. 8: Ablation study of RecastGS (a) and LayeredCGS (b). * denotes excluding the finetuning stage. We conduct ablation experiments on the Mip-NeRF 360 dataset. Numbers in () indicate BD-BR Gain over the baseline.

5 Conclusion & Limitation

We present an object-level scalable 3DGS compression framework consisting of RecastGS and LayeredCGS. RecastGS reorganizes pretrained 3DGS into a region-aware layered hierarchy, while LayeredCGS enables feed-forward compression and flexible, truncatable bitstreams. Extensive experiments demonstrate that the proposed framework achieves superior rate–distortion performance. Our current ROI study focuses on foreground/background quality allocation, while more complex preferences involving multiple objects, thin structures, or severe occlusions remain valuable directions for future work. Future work will also extend the framework to support more 3DGS formats, such as anchor-based and pixel-aligned formats.

Acknowledgement

The authors would like to acknowledge the support from NSFC (No. 62402427, U24B20154), Li Auto, Information Technology Center and State Key Lab of CAD&CG, and Zhejiang University Education Foundation Qizhen Scholar Foundation.

References

1. 3DG: G-PCC codec description v12. ISO/IEC JTC 1/SC 29/WG 7 N00271 (2022)
2. Bagdasarian, M.T., Knoll, P., Li, Y., Barthel, F., Hilsmann, A., Eisert, P., Morgenstern, W.: 3dgs. zip: A survey on 3d gaussian splatting compression methods. In: Computer Graphics Forum. vol. 44, p. e70078. Wiley Online Library (2025)
3. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: 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)
4. Bjøntegaard, G.: Calculation of average PSNR differences between rd-curves. In: ITU-T SG 16/Q6, 13th VCEG Meeting. document VCEG-M33 (April 2001)
5. Cai, C., Chen, L., Zhang, X., Gao, Z.: End-to-end optimized roi image compression. IEEE Transactions on Image Processing **29**, 3442–3457 (2019)
6. Chen, Y., Li, M., Wu, Q., Lin, W., Harandi, M., Cai, J.: Pcgs: Progressive compression of 3d gaussian splatting. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI) (2026), oral
7. Chen, Y., Wu, Q., Li, M., Lin, W., Harandi, M., Cai, J.: Fast feedforward 3d gaussian splatting compression. In: Proceedings of the 13th International Conference on Learning Representations (ICLR) (2025)
8. Chen, Y., Wu, Q., Lin, W., Harandi, M., Cai, J.: Hac: Hash-grid assisted context for 3d gaussian splatting compression. In: European Conference on Computer Vision. pp. 422–438. Springer (2024)
9. Chen, Y., Wu, Q., Lin, W., Harandi, M., Cai, J.: Hac++: Towards 100x compression of 3d gaussian splatting. IEEE Transactions on Pattern Analysis and Machine Intelligence **47**(11), 10210–10226 (2025). https://doi.org/10.1109/TPAMI.2025.3594066
10. Choi, S., Song, H., Kim, J., Kim, T., Do, H.: Click-gaussian: Interactive segmentation to any 3d gaussians. In: European Conference on Computer Vision. pp. 289–305. Springer (2024)
11. Deutsch, L.P.: Deflate compressed data format specification version 1.3. https://tools.ietf.org/html/rfc1951 (1996), rFC 1951
12. Di Sario, F., Renzulli, R., Grangetto, M., Sugimoto, A., Tartaglione, E.: Gode: Gaussians on demand for progressive level of detail and scalable compression. arXiv preprint arXiv:2501.13558 (2025)
13. Doukas, C., Maglogiannis, I.: Region of interest coding techniques for medical image compression. IEEE Engineering in medicine and Biology Magazine **26**(5), 29–35 (2007)
14. Fan, Z., Wang, K., Wen, K., Zhu, Z., Xu, D., Wang, Z., et al.: Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. Advances in neural information processing systems **37**, 140138–140158 (2024)

15. Fang, G., Wang, B.: Mini-splatting: Representing scenes with a constrained number of gaussians. In: European conference on computer vision. pp. 165–181. Springer (2024)
16. Fang, G., Wang, B.: Mini-splatting2: Building 360 scenes within minutes via aggressive gaussian densification. arXiv preprint arXiv:2411.12788 (2024)
17. Girish, S., Gupta, K., Shrivastava, A.: Eagles: Efficient accelerated 3d gaussians with lightweight encodings. In: European Conference on Computer Vision. pp. 54–71. Springer (2024)
18. Hadizadeh, H., Bajić, I.V.: Saliency-aware video compression. IEEE Transactions on Image Processing **23**(1), 19–33 (2013)
19. Hanson, A., Tu, A., Singla, V., Jayawardhana, M., Zwicker, M., Goldstein, T.: Pup 3d-gs: Principled uncertainty pruning for 3d gaussian splatting. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 5949–5958 (2025)
20. He, S., Jie, G., Wang, C., Zhou, Y., Hu, S., Li, G., Ding, H.: Refersplat: Referring segmentation in 3d gaussian splatting. arXiv preprint arXiv:2508.08252 (2025)
21. Hedman, P., Philip, J., Price, T., Frahm, J.M., Drettakis, G., Brostow, G.: Deep blending for free-viewpoint image-based rendering. ACM Transactions on Graphics (ToG) **37**(6), 1–15 (2018)
22. Hoang, T.M., Zhou, J.: Rclc: Roi-based joint conventional and learning video compression. arXiv preprint arXiv:2107.06492 (2021)
23. Hu, W., Chai, W., Hao, S., Cui, X., Wen, X., Hwang, J.N., Wang, G.: Pointmap association and piecewise-plane constraint for consistent and compact 3d gaussian segmentation field. arXiv preprint arXiv:2502.16303 (2025)
24. Hu, X., Wang, Y., Fan, L., Luo, C., Fan, J., Lei, Z., Li, Q., Peng, J., Zhang, Z.: Sagd: Boundary-enhanced segment anything in 3d gaussian via gaussian decomposition. arXiv preprint arXiv:2401.17857 (2024)
25. Huang, H., Huang, W., Yang, Q., Xu, Y., Li, Z.: A hierarchical compression technique for 3d gaussian splatting compression. In: ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1–5. IEEE (2025)
26. Jin, J., Xia, F., Ding, F., Zhang, X., Liu, M., Zhao, Y., Lin, W., Meng, L.: Customizable roi-based deep image compression. IEEE Transactions on Circuits and Systems for Video Technology (2025)
27. Jurca, M.B., Royen, R., Giosan, I., Munteanu, A.: Rt-gs2: Real-time generalizable semantic segmentation for 3d gaussian representations of radiance fields. arXiv preprint arXiv:2405.18033 (2024)
28. Kaur, M., Wasson, V.: Roi based medical image compression for telemedicine application. Procedia Computer Science **70**, 579–585 (2015)
29. Kaur, R., Rani, R.: Roi and non-roi based medical image compression techniques: a survey and comparative review. In: 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC). pp. 550–555. IEEE (2018)
30. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. **42**(4), 139–1 (2023)
31. Kim, J.Y., Yi, C.H., Kim, T.Y.: Roi-centered compression by adaptive quantization for sports video. IEEE Transactions on Consumer Electronics **56**(2), 951–956 (2010)
32. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4015–4026 (2023)

33. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics (ToG) **36**(4), 1–13 (2017)
34. Lee, J.C., Rho, D., Sun, X., Ko, J.H., Park, E.: Compact 3d gaussian representation for radiance field. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21719–21728 (2024)
35. Lee, S., Shu, F., Sanchez, Y., Schierl, T., Hellge, C.: Compression of 3d gaussian splatting with optimized feature planes and standard video codecs. arXiv preprint arXiv:2501.03399 (2025)
36. Li, Z., et al.: Roi-aware neural image compression. arXiv preprint arXiv:2303.16091 (2023)
37. Ling, L., Sheng, Y., Tu, Z., Zhao, W., Xin, C., Wan, K., Yu, L., Guo, Q., Yu, Z., Lu, Y., et al.: Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 22160–22169 (2024)
38. Liu, L., Chen, Z., Jiang, W., Wang, W., Xu, D.: Hemgs: A hybrid entropy model for 3d gaussian splatting data compression. arXiv preprint arXiv:2411.18473 (2024)
39. Liu, X., Wu, X., Zhang, P., Wang, S., Li, Z., Kwong, S.: Compgs: Efficient 3d scene representation via compressed gaussian splatting. In: Proceedings of the 32nd ACM International Conference on Multimedia. pp. 2936–2944 (2024)
40. Liu, Y., Zhong, Z., Zhan, Y., Xu, S., Sun, X.: Maskgaussian: Adaptive 3d gaussian representation from probabilistic masks. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 681–690 (2025)
41. Liu, Z., Song, R., Huang, Y., Hu, Y., Zhang, X., Shao, J., Lin, Z., Zhang, J.: Feed-forward 3d gaussian splatting compression with long-context modeling. arXiv preprint arXiv:2512.00877 (2025)
42. Lu, T., Yu, M., Xu, L., Xiangli, Y., Wang, L., Lin, D., Dai, B.: 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)
43. Lyu, W., Li, X., Kundu, A., Tsai, Y.H., Yang, M.H.: Gaga: Group any gaussians via 3d-aware memory bank. arXiv preprint arXiv:2404.07977 (2024)
44. Ma, Y., Zhai, Y., Yang, C., Yang, J., Wang, R., Zhou, J., Li, K., Chen, Y., Wang, R.: Variable rate roi image compression optimized for visual quality. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1936–1940 (2021)
45. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021)
46. Morgenstern, W., Barthel, F., Hilsmann, A., Eisert, P.: Compact 3d scene representation via self-organizing gaussian grids. In: European Conference on Computer Vision. pp. 18–34. Springer (2024)
47. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM transactions on graphics (TOG) **41**(4), 1–15 (2022)
48. Navaneet, K., Meibodi, K., Koohpayegani, S., Pirsiavash, H.: Compact3d: Compressing gaussian splat radiance field models with vector quantization. arxiv 2023. arXiv preprint arXiv:2311.18159
49. Niedermayr, S., Stumpfegger, J., Westermann, R.: 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)

50. Niemeyer, M., Manhardt, F., Rakotosaona, M.J., Oechsle, M., Duckworth, D., Gosula, R., Tateno, K., Bates, J., Kaeser, D., Tombari, F.: Radsplat: Radiance field-informed gaussian splatting for robust real-time rendering with 900+ fps. In: 2025 International Conference on 3D Vision (3DV). pp. 134–144. IEEE (2025)
51. Papantonakis, P., Kopanas, G., Kerbl, B., Lanvin, A., Drettakis, G.: Reducing the memory footprint of 3d gaussian splatting. Proceedings of the ACM on Computer Graphics and Interactive Techniques **7**(1), 1–17 (2024)
52. Qin, M., Li, W., Zhou, J., Wang, H., Pfister, H.: Langsplat: 3d language gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20051–20060 (2024)
53. de Queiroz, R.L., Chou, P.A.: Compression of 3d point clouds using a region-adaptive hierarchical transform. IEEE Transactions on Image Processing **25**, 3947–3956 (2016)
54. Ramlot, B., Courteaux, M., Lambert, P., Van Wallendael, G.: Potr: Post-training 3dgs compression. arXiv preprint arXiv:2601.14821 (2026)
55. Sandri, G., Figueiredo, V.F., Chou, P.A., De Queiroz, R.: Point cloud compression incorporating region of interest coding. In: 2019 IEEE International Conference on Image Processing (ICIP). pp. 4370–4374. IEEE (2019)
56. Schnabel, R., Klein, R.: Octree-based Point-Cloud Compression. In: Symposium on Point-Based Graphics. The Eurographics Association (2006). `https://doi.org/10.2312/SPBG/SPBG06/111-120`
57. Song, Z., Fu, J., Zhang, J., Lu, X., Jia, C., Ma, S., Gao, W.: Tinysplat: Feed-forward approach for generating compact 3d scene representation. arXiv preprint arXiv:2506.09479 (2025)
58. Tang, H., Liu, Z., Li, X., Lin, Y., Han, S.: Torchsparse: Efficient point cloud inference engine. Proceedings of Machine Learning and Systems **4**, 302–315 (2022)
59. Tran, H.N., Di Sario, F., Spadaro, G., Valenzise, G., Tartaglione, E.: Rave: Rate-adaptive visual encoding for 3d gaussian splatting. arXiv preprint arXiv:2512.07052 (2025)
60. Wang, H., Zhu, H., He, T., Feng, R., Deng, J., Bian, J., Chen, Z.: End-to-end rate-distortion optimized 3d gaussian representation. In: European Conference on Computer Vision. pp. 76–92. Springer (2024)
61. Wang, Y., Li, Z., Guo, L., Yang, W., Kot, A., Wen, B.: Contextgs: Compact 3d gaussian splatting with anchor level context model. Advances in neural information processing systems **37**, 51532–51551 (2024)
62. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)
63. Xie, L., Gao, W., Zheng, H., Li, G.: Roi-guided point cloud geometry compression towards human and machine vision. In: Proceedings of the 32nd ACM International Conference on Multimedia. pp. 3741–3750 (2024)
64. Xie, S., Zhang, W., Tang, C., Bai, Y., Lu, R., Ge, S., Wang, Z.: Mesongs: Post-training compression of 3d gaussians via efficient attribute transformation. In: European Conference on Computer Vision. Springer (2024)
65. Yan, M., Zhang, R., Xiao, X., Wang, W.: Detvpcc: Roi-based point cloud sequence compression for 3d object detection. arXiv preprint arXiv:2502.04804 (2025)
66. Yang, Q., Yang, K., Xing, Y., Xu, Y., Li, Z.: A benchmark for gaussian splatting compression and quality assessment study. In: Proceedings of the 6th ACM International Conference on Multimedia in Asia. pp. 1–8 (2024)

67. Yang, Y., Yuan, H., Wang, A.: An roi-aware lidar point cloud compression method in autonomous driving. In: 2025 IEEE International Conference on Vehicular Electronics and Safety (ICVES). pp. 284–289. IEEE (2025)
68. Ye, M., Danelljan, M., Yu, F., Ke, L.: Gaussian grouping: Segment and edit anything in 3d scenes. In: European conference on computer vision. pp. 162–179. Springer (2024)
69. Ying, H., Yin, Y., Zhang, J., Wang, F., Yu, T., Huang, R., Fang, L.: Omniseg3d: Omniversal 3d segmentation via hierarchical contrastive learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20612–20622 (2024)
70. You, K., Chen, T., Ding, D., Asif, M.S., Ma, Z.: Reno: Real-time neural compression for 3d lidar point clouds. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 22172–22181 (2025)
71. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018)
72. Zhang, W., Zhao, Y., Wang, Q., Song, L., Cheng, Z.: D-fcgs: Feedforward compression of dynamic gaussian splatting for free-viewpoint videos. In: Proceedings of the AAAI Conference on Artificial Intelligence (2026)
73. Zhang, Y., Jia, W., Niu, W., Yin, M.: Gaussianspa: An" optimizing-sparsifying" simplification framework for compact and high-quality 3d gaussian splatting. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 26673–26682 (2025)
74. Zhao, Y., Xu, W., Zheng, R., Qiao, P., Liu, C., Chen, J.: isegman: Interactive segment-and-manipulate 3d gaussians. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 661–670 (2025)
75. Zhou, S., Chang, H., Jiang, S., Fan, Z., Zhu, Z., Xu, D., Chari, P., You, S., Wang, Z., Kadambi, A.: Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21676–21685 (2024)
76. Zhu, R., Qiu, S., Liu, Z., Hui, K.H., Wu, Q., Heng, P.A., Fu, C.W.: Rethinking end-to-end 2d to 3d scene segmentation in gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3656–3665 (2025)