This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

CF^3: Compact and Fast 3D Feature Fields

Hyunjoon Lee Joonkyu Min Jaesik Park*
Seoul National University, Republic of Korea
{hjlee4772, timothy0609, jaesik.park}@snu.ac.kr

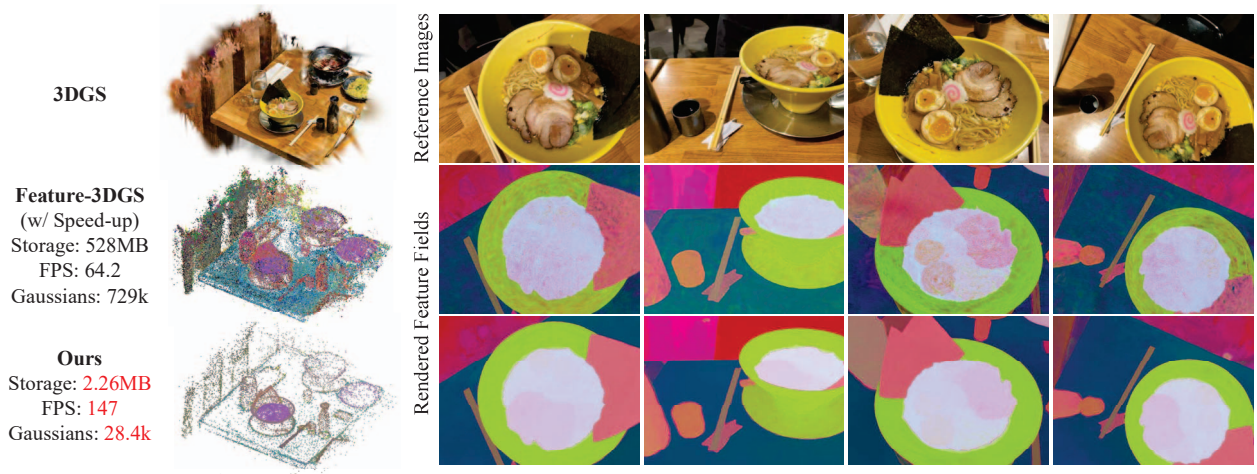

Figure 1. We propose CF^3 for constructing a compact and fast 3D feature field from 3D Gaussians. The previous method (Feature-3DGS) jointly optimizes features with colors, resulting in excessive Gaussians for rendering the feature field. CF^3 effectively compresses and sparsifies the 3D feature field while maintaining sufficient details as shown in the rendered feature maps.

Abstract

3D Gaussian Splatting (3DGS) has begun incorporating rich information from 2D foundation models. However, most approaches rely on a bottom-up optimization process that treats raw 2D features as ground truth, incurring increased computational costs. We propose a top-down pipeline for constructing compact and fast 3D Gaussian feature fields, namely, CF^3. We first perform a fast weighted fusion of multi-view 2D features with pre-trained Gaussians. This approach enables training a per-Gaussian autoencoder directly on the lifted features, instead of training autoencoders in the 2D domain. As a result, the autoencoder better aligns with the feature distribution. More importantly, we introduce an adaptive sparsification method that optimizes the Gaussian attributes of the feature field while pruning and merging the redundant Gaussians, constructing an efficient representation with preserved geometric details. Our approach achieves a competitive 3D feature field using as little as 5% of the Gaussians compared to Feature-3DGS.

*Corresponding author.

1. Introduction

Recent advances in 3D scene reconstruction have achieved significant progress in rendering high-fidelity images and precise 3D models, as exemplified by methods such as NeRF [31] and 3DGS [18]. With these advances, modern methods have aimed to integrate rich information from 2D foundation models, like CLIP [36], LSeg [25], and SAM [20] into 3D representations. These methods extract patch-level or pixel-level features from multi-view images, including those designed for semantic understanding. In the case of semantic features, the extracted representations are distilled into the 3D space, forming a language or semantic 3D field capable of handling open-vocabulary queries, *e.g.*, 'wall', 'sofa', 'chair', in real-time.

Prior works in this category [35, 56] typically optimize the embedding of semantic features, akin to learning color via photometric loss, across all Gaussians using multi-view raw visual feature maps. Since this joint color and feature learning strategy forces the recovery of color details with an excessive number of Gaussians, the resulting feature fields are often heavy and redundant. Furthermore, directly embedding high-dimensional language features into 3D Gaussians

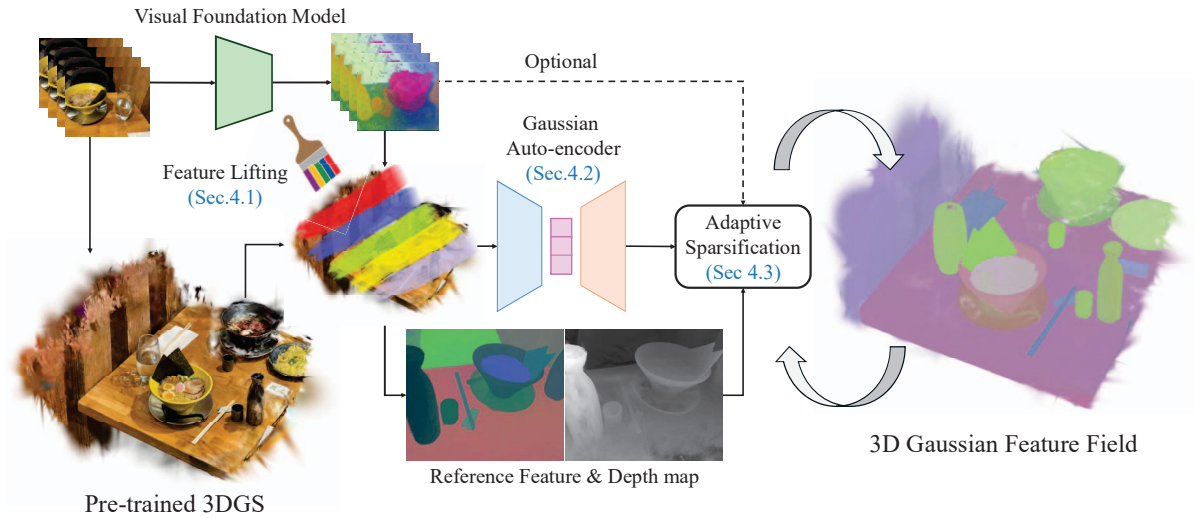

Figure 2. **Overview of our CF^3 pipeline.** We utilize pre-trained 3D Gaussians to construct a 3D feature field. We adopt a weighted-sum strategy to lift features extracted from a visual foundation model into 3D. Subsequently, a per-Gaussian autoencoder compresses high-dimensional features into lower-dimensional embeddings, effectively removing noisy features through a variance filtering step. Afterward, adaptive sparsification merges redundant Gaussians, efficiently reducing the total Gaussian count and resulting in a compact 3D feature field.

incurs significant storage and computational costs. Several methods have been proposed to address these issues. For example, feature compression using autoencoders [35] or decoder-only reconstruction [56], as well as hash-grid techniques [57] and vector quantization [40], have been explored. However, these methods [35, 40, 56, 57] do not explicitly consider that Gaussians optimized for color may be redundant for expressing a feature field. In addition, previous feature embedding methods [26, 35, 40, 48, 56, 57] rely on raw features from 2D foundation models, which often lack multi-view consistency [5, 9].

We propose an approach to eliminate redundant Gaussians and achieve high-quality feature fields. An overview of the compactness of our method is shown in Fig. 1. Figure 2 provides an overview of our pipeline, illustrating the stages of feature lifting, compression, and adaptive sparsification.

Similar to 3D-aware training in FiT3D [50] and CONDENSE [52], we first compute a weighted combination of 2D features in 3D, namely feature lifting, with a pre-trained 3DGS. This scheme quickly achieves feature quality comparable to results from approaches that jointly optimize images and features. We employ these spatially coherent and view-consistent rendered features as reference features.

Moreover, unlike Feature-3DGS [56] and LangSplat [35] that learn a per-pixel decoder, we suggest lifting the feature first (to get reference features) and then compressing it using a per-Gaussian autoencoder. Since each Gaussian is directly assigned a fused and view-consistent reference feature, our method avoids the need for pre-compression of 2D feature maps, enabling direct training of the autoencoder for each Gaussian. Combined with variance filtering, this approach effectively removes inaccurate features that may arise during the lifting process, ensuring more reliable feature extraction.

Building on this compression, we propose an adaptive sparsification process to optimize the Gaussian feature field even further. This step optimizes Gaussian attributes and merges redundant Gaussians in stable regions. Here, stable regions refer to areas with a small gradient that already represent the scene well, making further refinement unnecessary. We summarize our main contributions below:

- We build a compact 3D feature field by lifting features via a pre-trained 3DGS and compressing them with a per-Gaussian autoencoder. This ensures robustness across downstream tasks since each Gaussian directly encodes view-consistent reference features.
- Our adaptive sparsification step optimizes the Gaussian feature field even further, which involves pruning and merging redundant Gaussians, while preserving essential details. As a result, our method achieves competitive performance while using as little as 5% of the original number of Gaussians, improving storage efficiency and rendering speed.

2. Related Work

2.1. Visual feature embedding with NeRF

Neural Radiance Fields (NeRF)-based approach pioneered beyond basic scene reconstruction by incorporating high-dimensional features extracted from 2D vision foundation models into 3D representations. By embedding features in NeRF, tasks such as semantic segmentation, object decom-

position, language-based querying, and editing are enabled.

These feature-embedded approaches can be broadly categorized into three groups. First, some approaches distill large-scale 2D embeddings (e.g., CLIP, DINO) into 3D fields for open-vocabulary queries or text-driven object segmentation [10, 18, 21, 28, 44]. They typically employ multi-scale patch extraction or pixel-aligned semantic features [15, 25], combined with feature alignment and additional losses (e.g., regularization) to enhance geometry and segmentation. Second, several approaches introduce object-level decomposition or local NeRF blocks for scene editing or refining sparse/noisy 2D annotations [45, 51, 54]. They achieve higher interpretability and efficient object manipulation through targeted object fields, specialized losses, or local MLPs. Finally, some methods address panoptic labeling in 3D by fusing bounding primitives or 2D panoptic masks with NeRF [14, 22, 42].

2.2. Visual feature embedding with 3DGS

3D Gaussian splatting [17] has demonstrated strong performance in real-time novel view rendering by representing scenes explicitly via anisotropic Gaussians, which can be rasterized and blended at high speed. To further enhance these representations with semantic information, several works [35, 56, 57] have proposed integrating features from 2D foundation models. Early efforts employ optimization-based feature distillation, where embeddings (e.g., from CLIP, DINO, or SAM) are lifted into 3D space through iterative optimization. Subsequent approaches [26, 40, 48] address the storage overhead of large embeddings by quantizing or compressing features, or by clustering Gaussians into superpoint-like structures. A few methods adopt training-free schemes, aggregating 2D features into 3D with a weighted average method rather than explicit backpropagation [7, 30]. Others [12, 47] attempt a feedforward model that can process sparse or unposed images and generate a feature-embedded Gaussians in a single pass.

2.3. Reducing storage overhead

Recent research on 3DGS focuses on reducing storage overhead while maintaining quality through three complementary strategies: (1) compressing individual Gaussian attributes through vector quantization or selective spherical harmonics (SH) pruning [11, 34, 46], (2) reorganizing scenes with structured encodings (anchor-based or hash-grid-based) to leverage spatial coherence [4, 23, 29, 41], and (3) adaptively controlling splat density by pruning less significant Gaussians or densifying under-reconstructed regions [8, 16, 38, 46, 53]. In attribute compression, highly correlated features, such as scale, rotation, or SH color coefficients, are typically clustered into codebooks to reduce redundancy, while low-bit quantization and optional re-encoding with standard codecs further reduce storage requirements [32, 41, 49]. Structured representations organize splats in anchor-based clusters, 2D grids, employ octrees, or Morton ordering to efficiently skip empty regions [13, 19, 32, 49], sometimes replacing SH with learned MLPs [23]. Pruning strategies eliminate overlapping or negligible splats [4, 11, 16, 23, 46], while selective densification enhances fine details using multi-view gradients, as well as depth or normal cues [8, 38, 53].

3. Preliminary

3DGS scene $\mathcal{S} = \{g_i | i = 1, \cdots, N\}$ is represented with N Gaussians, where each Gaussian has a center coordinate $\boldsymbol{\mu} \in \mathbb{R}^3$, a covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$, an opacity $\alpha \in \mathbb{R}_+$.

$$g_i(\mathbf{x}) = \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right). \quad (1)$$

Color $\boldsymbol{c}$ at each pixel in the image is rendered via alpha blending of Gaussian's color or spherical harmonics features $\boldsymbol{c}_i$ considering depth order to the viewpoint. Similarly, depth is rendered by weighting each Gaussian with distance d_i, defined as the distance from the camera center to each Gaussian [17].

$$C = \sum_{i=1}^{N} \boldsymbol{c}_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j) = \sum_{i=1}^{N} \boldsymbol{c}_i \alpha_i T_i = \sum_{i=1}^{N} \boldsymbol{c}_i w_i, \quad (2)$$

$$D = \sum_{i=1}^{N} d_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j) = \sum_{i=1}^{N} d_i \alpha_i T_i = \sum_{i=1}^{N} d_i w_i, \quad (3)$$

where $T_i \in \mathbb{R}_+$ is transmittance. We denote w_i as the *weight of the corresponding Gaussian* contributing to each pixel.

4. Method

4.1. Feature Lifting

The prior methods optimize the features during 3DGS [56, 57] training, which result in a long training time, making it difficult to scale up. We use an alternative and scalable solution to lift visual features to our 3DGS scene. Given M images, P pixels each, let's assume we have image features $\boldsymbol{F}_{m,p}$ for p-th pixel in m-th image, where $\|\boldsymbol{F}_{m,p}\| = 1$. Let $\mathcal{G}_{m,p}$ be an index set of Gaussians that are projected onto pixel p of image m.

The problem is to minimize the gap between the image features $\boldsymbol{F}_{m,p}$ and the rendered features $\sum_{i\in\mathcal{G}_{m,p}} w_{i,m,p}\boldsymbol{f}_i$, $\boldsymbol{f}_i$ indicates corresponding features for each 3D Gaussians with the constraint of $\|\boldsymbol{f}_i\| = 1$. Here, $w_{i,m,p}$ refers to the weight introduced in Eq. (2). The approximate solution is simply computing the weighted sum over a set of pixels that are included in a 2D splat of Gaussian g_i, noted as $\mathcal{P}_{i,m}$:

$$\boldsymbol{f}_i \approx \frac{\sum_{m=1}^{M}\sum_{p\in\mathcal{P}_{i,m}} w_{i,m,p}\boldsymbol{F}_{m,p}}{\sum_{m=1}^{M}\sum_{p\in\mathcal{P}_{i,m}} w_{i,m,p}}. \quad (4)$$

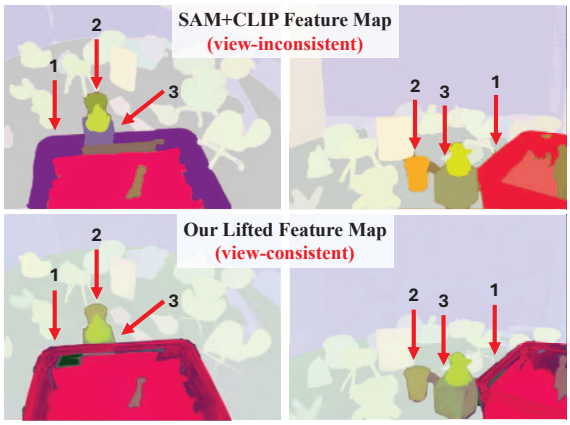

Figure 3. **Feature lifting.** The raw features from visual foundation models are not view-consistent. Feature lifting (Sec. 4.1) alleviates this inconsistency.

This idea appears in recent training-free feature aggregation methods [7, 30]. As shown in Fig. 3, lifting visual features to 3D Gaussians can reduce multi-view inconsistencies [50].

In addition, we can measure the variance of the approximated features. Without considering the covariance among feature dimensions, the variance of each d-th dimension of features can be computed as follows:

$$\mathrm{Var}(\boldsymbol{f}_i)_d \approx \frac{\sum_{m=1}^{M}\sum_{p\in\mathcal{P}_{i,m}} w_{i,m,p}(\boldsymbol{F}_{m,p})_d^2}{\sum_{m=1}^{M}\sum_{p\in\mathcal{P}_{i,m}} w_{i,m,p}} - (\boldsymbol{f}_i)_d^2. \quad (5)$$

Most of the Gaussians at accurate positions with consistent features have low variance. However, some 3D Gaussians with inaccurate geometry or those located at the edges of objects often average irrelevant information. Therefore, we filter out i-th Gaussian whose norm of the approximated variance $\mathrm{Var}(\boldsymbol{f}_i)$ is larger than the top 0.01% for the downstream pipeline.

4.2. Feature Compression

Unlike the existing method [35] that trains an autoencoder before feature lifting, we suggest lifting the features first and then compressing them using a per-Gaussian autoencoder.

As shown in Fig. 3, our autoencoder is trained directly on the lifted features, making it better aligned with the actual feature distribution used during inference. Note that our autoencoder (MLP with five layers: [128, 64, 32, 16, 3]-*dims* for encoding) compresses the D-dimensional lifted features $\boldsymbol{f}$ into a just 3-dimensional latent space. Interestingly, this is equivalent to treating the encoded feature as 3-channel RGB colors. This design allows us to leverage the existing 3DGS rasterizer directly, and the outputs can be directly decoded into semantic features. Our autoencoder is trained with MSE loss, together with cosine-similarity loss and a lightweight similarity structure preserving regularizer.

The objective is defined as follows:

$$\mathcal{L} = \mathcal{L}_{MSE} + \lambda_{cos}\cdot\mathcal{L}_{cos} + \lambda_{struc}\cdot\mathcal{L}_{struc}, \quad (6)$$

$$\mathcal{L}_{MSE} = \mathbb{E}_{i\in\mathcal{G}}\left[\|\mathcal{D}(\mathcal{E}(\boldsymbol{f}_i)) - \boldsymbol{f}_i\|_2\right], \quad (7)$$

$$\mathcal{L}_{cos} = \mathbb{E}_{i\in\mathcal{G}}\left[1 - \frac{\langle\mathcal{D}(\mathcal{E}(\boldsymbol{f}_i)),\ \boldsymbol{f}_i\rangle}{\|\mathcal{D}(\mathcal{E}(\boldsymbol{f}_i))\|\cdot\|\boldsymbol{f}_i\|}\right], \quad (8)$$

$$\mathcal{L}_{struc} = \mathbb{E}_{i\neq j,(i,j)\in\mathcal{G}}\left[\left|\cos(\boldsymbol{f}_i,\boldsymbol{f}_j) - \cos(\mathcal{E}(\boldsymbol{f}_i),\mathcal{E}(\boldsymbol{f}_j))\right|\right], \quad (9)$$

where $\mathcal{G}$ is a set of gaussians, $\mathcal{E}(\boldsymbol{f}_i)$ is the encoded latent feature, and $\mathcal{D}(\mathcal{E}(\boldsymbol{f}_i))$ is the corresponding reconstruction.

4.3. Adaptive Sparsification

As a next step, we optimize the Gaussian attributes $(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, \boldsymbol{f})$ in our Gaussian feature field, which involves iterative pruning and merging 3D Gaussians to reduce redundancy. Figure 4 shows the overall sparsification pipeline. The sparsification process uses the rendered reference feature $\boldsymbol{F}_{ref}$ and depth map D_{ref} of the Gaussian feature field being optimized. They are obtained from Eqs. (2) and (3). Note that we can reuse Eq. (2) for feature rendering since the compressed feature is 3-dimensional. These view-consistent features stabilize optimization by providing supervision across views. The depth regularization term encourages geometric consistency with the original scene structure, enabling better alignment between the pre-trained 3DGS and CF^3.

We define the objective for optimizing our 3D Gaussian attributes as follows:

$$\mathcal{L} = \mathcal{L}_f + \lambda_{depth}\cdot\mathcal{L}_{depth}, \quad (10)$$

$$\mathcal{L}_f = \|\boldsymbol{F}_{ref} - \boldsymbol{F}\|_1, \quad (11)$$

$$\mathcal{L}_{depth} = \|D_{ref} - D\|_1, \quad (12)$$

where $\boldsymbol{F}$ is the rendered feature map followed by the trained decoder.

Our Gaussian field optimization involves the following adaptive sparsification steps. (1) *Pruning*. By following LightGaussian [11], we prune the 3D Gaussians based on the global contribution $C(g_i)$, which is the sum of the weights on each image pixel:

$$C(g_i) = \sum_{m=1}^{M}\sum_{p\in\mathcal{P}_{i,m}} w_{i,m,p}. \quad (13)$$

(2) *Merging*. We then iteratively merge the neighboring pairs of Gaussians with the same semantic information. For each Gaussian, we identify its k-nearest neighbors and then choose pairs with a significant overlap. We measure the overlap between neighboring i-th and j-th 3D Gaussians using Mahalanobis distance d_M,

$$d_M = (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^\top\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) < \chi^2_\beta, \quad (14)$$

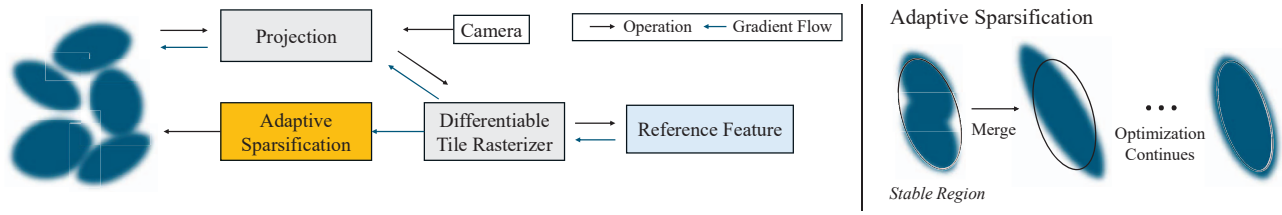

Figure 4. **Overview of our adaptive sparsification pipeline.** Unlike the original 3D Gaussian Splatting, which preserves fine-grained details for photorealistic rendering, our method focuses on feature field reconstruction and merges redundant Gaussians to reduce unnecessary density, achieving effective sparsification.

Algorithm 1 Adaptive Sparsification

$\boldsymbol{P}, \boldsymbol{S}, A \leftarrow$ Initial 3DGS position, scale, opacity
$\boldsymbol{F} \leftarrow$ lifted feature on 3DGS ▷ Sec. 4.1
Freeze feature lifted 3DGS $(\boldsymbol{P}, \boldsymbol{S}, \boldsymbol{F}, A)$
$\boldsymbol{P}_C, \boldsymbol{S}_C, A_C \leftarrow \boldsymbol{P}, \boldsymbol{S}, A$ ▷ initialize CF^3
Enc, **Dec** ▷ pretrained from lifted features $\boldsymbol{F}$
$\boldsymbol{C}_C \leftarrow$ **Enc**$(\boldsymbol{F})$ ▷ Encode features to color channels
$i \leftarrow 0$ ▷ Iteration Count
while $i <$ MaxIteration **do**
  $\boldsymbol{V} \leftarrow$ SampleTrainingView()
  $\boldsymbol{I} \leftarrow$ Rasterize$(\boldsymbol{P}_C, \boldsymbol{S}_C, \boldsymbol{C}_C, A_C, \boldsymbol{V})$
  $\boldsymbol{F}_{ref} \leftarrow$ RasterizeFeature$(\boldsymbol{P}, \boldsymbol{S}, \boldsymbol{F}, A, \boldsymbol{V})$
  $\mathcal{L} \leftarrow$ Loss(**Dec**$(\boldsymbol{I}), \boldsymbol{F}_{ref})$
  $(\boldsymbol{P}_C, \boldsymbol{S}_C, \boldsymbol{C}_C, A_C) \leftarrow$ Adam(∇L)
  if IsPruneIteration(i) **then**
    PRUNEGAUSSIANS
  if IsMergeIteration(i) **then**
    MERGEGAUSSIANS
  $i \leftarrow i + 1$

function PRUNEGAUSSIANS
  for all Gaussians $g(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\alpha}, \boldsymbol{c})$ **do**
    if $C(g) < \tau_{con}$ **then** ▷ global contribution Eq. 13
      PrunePoints(g) ▷ prune gaussians

function MERGEGAUSSIANS
  for all Gaussians $g_i(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \alpha_i, \boldsymbol{c}_i)$ **do**
    if $\nabla L < \tau_{grad}$ **then** ▷ gradient threshold
      for all k-neighbors $g_j(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \alpha_j, \boldsymbol{c}_j)$ **do** ▷ $j \neq i$
        $\boldsymbol{d} \leftarrow \boldsymbol{\mu}_j - \boldsymbol{\mu}_i$ ▷ distance between Gaussians
        $d_M \leftarrow \max(\boldsymbol{d}^T \boldsymbol{\Sigma}_i^{-1} \boldsymbol{d},\ \boldsymbol{d}^T \boldsymbol{\Sigma}_j^{-1} \boldsymbol{d})$
        if $\langle \boldsymbol{c}_i, \boldsymbol{c}_j \rangle > \tau_{sim}$ **and** $d_M < \chi^2_\beta$ **then**
          MergeGaussian(g_i, g_j) ▷ Equation (18)

which effectively quantifies the separation of two Gaussian distributions relative to their covariance, and uses it for deciding the pairs to be merged.

Inspired by moment matching method for Gaussian mixture reduction [39], the new attributes for 3D feature Gaussians $(\boldsymbol{\mu}_{new}, \boldsymbol{\Sigma}_{new}, \alpha_{new}, \boldsymbol{f}'_{new})$ that approximately represents the two overlapping Gaussians $(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \alpha_i, \boldsymbol{f}'_i)$, $(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \alpha_j, \boldsymbol{f}'_j)$ are computed by the following equation:

$$\boldsymbol{\mu}_{new} = \frac{\alpha_i \boldsymbol{\mu}_i + \alpha_j \boldsymbol{\mu}_j}{\alpha_i + \alpha_j}, \tag{15}$$

$$\boldsymbol{\Sigma}_{new} = \frac{\alpha_i \left(\boldsymbol{\Sigma}_i + \boldsymbol{\mu}_i \boldsymbol{\mu}_i^\top\right) + \alpha_j \left(\boldsymbol{\Sigma}_j + \boldsymbol{\mu}_j \boldsymbol{\mu}_j^\top\right)}{\alpha_i + \alpha_j} - \boldsymbol{\mu}_{new} \boldsymbol{\mu}_{new}^\top, \tag{16}$$

$$\alpha_{new} = \alpha_i + \alpha_j - \alpha_i \alpha_j, \tag{17}$$

$$\boldsymbol{f}'_{new} = \frac{\alpha_i \boldsymbol{f}'_i + \alpha_j \boldsymbol{f}'_j}{\alpha_i + \alpha_j}, \tag{18}$$

where $\boldsymbol{f}' = \mathcal{E}(\boldsymbol{f}) \in \mathbb{R}^3$ denotes the latent feature compressed by the autoencoder $\mathcal{E}$. Through our adaptive sparsification step, we construct a compact 3D feature field with significantly fewer Gaussians than the original 3DGS. The algorithm is summarized in Algorithm 1.

5. Experiments

To evaluate our method, we conducted comparative experiments with other state-of-the-art feature-embedded 3D Gaussian splatting methods. Further, we demonstrate the effectiveness of the feature-wise weighted averaging approach by applying it to both 3DGS [17] and LightGaussian [11]. We evaluate our method by measuring storage efficiency and performance on downstream tasks, including semantic segmentation and localization.

5.1. Setup

We use the widely adopted Replica [43] and LERF [18] datasets. We evaluate semantic segmentation on the Replica dataset using LSeg [25] and MaskCLIP [55] across four scenes used by Feature-3DGS: room 0, room 1, office 3, and office 4. Feature-3DGS [56] can embed the original feature directly into the 3D Gaussian splatting framework. It trains a computationally efficient 1×1 decoder, a lower-dimensional feature can also be embedded into the 3D Gaussian splatting framework with minimal performance loss. We conducted experiments on Feature-3DGS with original, 128-dimensional, and 3-dimensional features to compare them

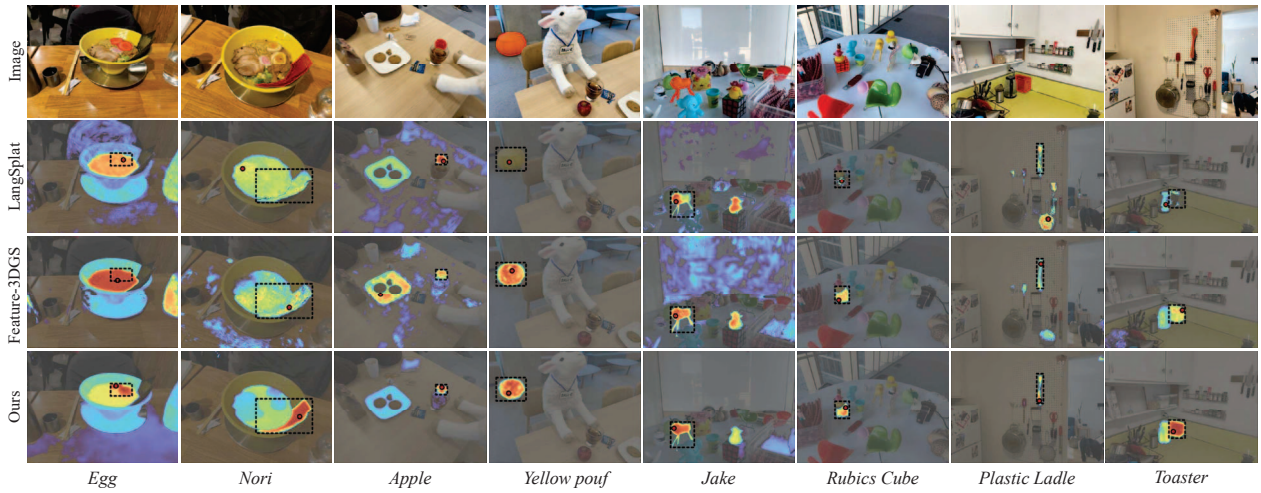

Figure 5. **Qualitative comparison.** We visualize open-vocabulary semantic segmentation and localization results using CLIP [36] with SAM [20] features on the LERF [18] dataset. Our method shows precise results even for small objects in these tasks. Feature-3DGS [56] is tested with the speed-up module *(128 dim)*. We overlay the ground truth segmentation for the query in red on the image for visualization.

	Storage↓	FPS↑	mIoU↑	Acc.↑	#G↓
Feature-3DGS (512)	1393.9M	7.2	73.0	91.9	636k
Feature-3DGS (128)	463.9M	113.8	**73.4**	**92.9**	640k
Feature-3DGS (3)	160.8M	198.8	21.3	59.2	644k
3DGS*	1336.2M	6.8	70.1	90.9	600k
LightGaussian*	458.8M	7.3	70.0	91.0	204k
CF^3 (Ours)	3.6M	**328.3**	70.8	91.6	**47k**
CF^3 + VQ (Ours)	**1.7M**	327.3	70.1	90.9	**47k**

Table 1. **Evaluation on Relica dataset with LSeg [25].** The asterisk (*) denotes results with feature lifting.

	Storage↓	FPS↑	mIoU↑	Acc.↑	#G↓
LangSplat	314.9M	33.4	44.7	72.3	1270k
Feature-3DGS (128)	1031.7M	55.6	53.8	75.8	1423k
Feature-3DGS (3)	345.6M	90.6	4.3	3.5	1394k
3DGS*	2832.8M	1.7	**56.7**	**85.4**	1289k
LightGaussian*	986.0M	1.8	55.2	83.9	438k
CF^3 (Ours)	4.2M	**145.0**	52.4	76.8	**55k**
CF^3 + VQ (Ours)	**1.9M**	144.3	51.7	75.7	**55k**

Table 2. **Evaluation on LERF dataset with CLIP + SAM [20, 36].**

with our compact and efficient representation. We then rendered the embedded features and computed similarity with text queries to obtain segmentation masks after thresholding. We measured the mean intersection-over-union (mIoU) and accuracy following the evaluation protocol [56]. For the LERF [18] dataset, we followed the LERF evaluation protocol and assessed mIoU and localization accuracy for four scenes: Ramen, Figurines, TeaTime, and Waldo Kitchen. We use the semantic features from LangSplat [35] in this experiment. Since CLIP [36] provides image-level features rather than pixel-level, LangSplat uses SAM [20] to extract region-specific CLIP features. These features are divided into whole, part, and subpart levels. Since our focus is on evaluating feature representations rather than the feature map granularity, we used the whole-level feature map consistently across all methods for a fair comparison.

On top of our method, we apply additional vector quantization following LightGaussian [11] to compress the feature field even further. We employed 3D Gaussian splatting scenes trained with 30k iterations. The same setup applies to Feature-3DGS and LangSplat in all experiments, including FPS measurements, conducted on a single NVIDIA RTX6000 Ada GPU.

5.2. Comparison

We conduct comparisons between CF^3 and Feature-3DGS, using LSeg features on the Replica dataset. As shown in Tab. 1, our CF^3 achieves competitive mIoU and accuracy while providing 121× more compact 3D feature field than Feature-3DGS with a speed-up module. By employing adaptive sparsification to merge and prune unnecessary Gaussians, CF^3 achieves comparable performance using fewer than 10% of the Gaussians. Additional vector quantization (CF^3+VQ) results in an even more compact 3D feature field, without notable performance degradation. In this experiment, we also incorporate the raw feature map as regularization.

We then compare LangSplat, Feature-3DGS, and CF^3 using the LERF dataset. We adopt the same feature map used in LangSplat. LangSplat compresses the 512-dimensional features to 3 dimensions using an autoencoder before lifting them to 3DGS, resulting in a more compact representation than Feature-3DGS. In contrast, our per-Gaussian autoencoder, trained under the same feature distribution, leads to cleaner segmentation, as shown in Fig. 5. Consequently,

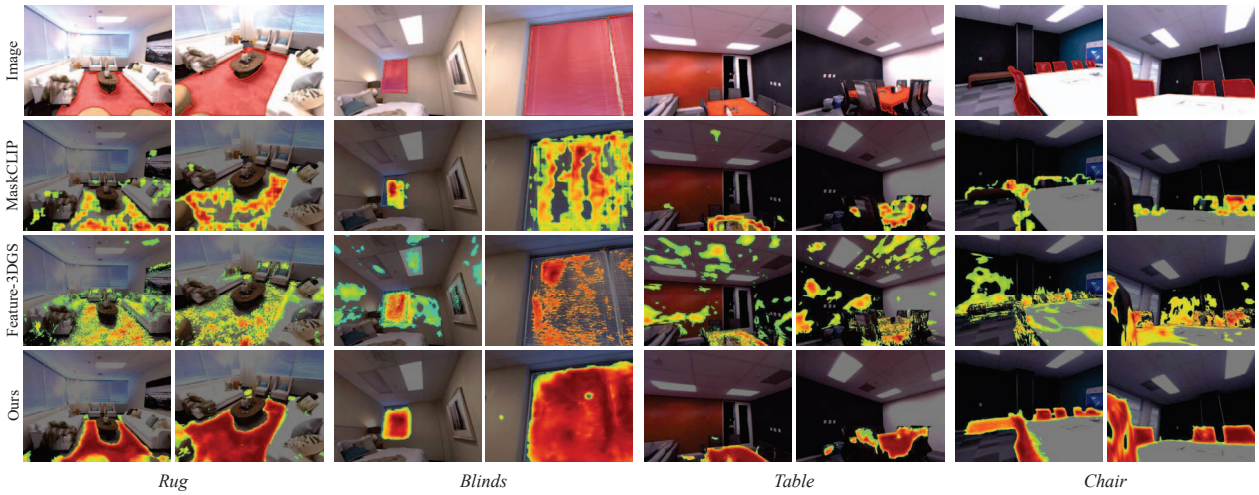

Figure 6. **Qualitative comparison.** We visualize open-vocabulary semantic segmentation results using MaskCLIP [55] features. Feature-3DGS [56] is tested with the speed-up module *(128 dim)*. We highlight the ground truth masks in red for the corresponding query texts.

Config			MaskCLIP (Replica)				LSeg (Replica)					CLIP+SAM (LERF)				
VF	P	M	Storage↓	FPS↑	mIoU↑	#G↓	Storage↓	FPS↑	mIoU↑	Acc↑	#G↓	Storage↓	FPS↑	mIoU↑	Acc↑	#G↓
-	-	-	42.5M	245	42.8	600k	42.6M	254	61.0	87.6	600k	90.7M	101	29.7	57.8	1289k
✓	✓	-	11.5M	311	46.6	152k	12.1M	318	71.0	92.0	165k	23.4M	130	53.4	77.9	324k
✓	-	✓	27.6M	279	46.3	384k	25.6M	238	70.9	92.1	355k	20.4M	139	52.4	77.2	284k
-	✓	✓	3.0M	335	46.0	36k	3.4M	324	69.8	91.5	43k	4.2M	143	54.5	74.4	56k
✓	✓	✓	2.6M	341	46.9	29k	3.6M	328	70.8	91.6	47k	4.2M	145	52.2	76.8	55k

Table 3. **Ablation study across all experiments.** Variance Filtering (VF), Pruning (P), Merging (M), First , Second , Third

Metrics	Storage↓	FPS↑	mIoU↑	#G↓
MaskCLIP	-	-	29.3	-
Feature-3DGS (512)	1443.3M	7.2	35.9	758k
Feature-3DGS (128)	474.8M	118.3	33.7	760k
Feature-3DGS (3)	162.3M	198.5	18.4	760k
3DGS*	1348.5M	7.2	46.3	600k
LightGaussian*	448.3M	7.4	46.2	204k
CF^3 (Ours)	2.6M	340.5	46.9	**29k**
CF^3+VQ (Ours)	**1.5M**	**342.3**	**47.1**	**29k**

Table 4. **Evaluation on Replica dataset with MaskCLIP [55].**

as Tab. 2 indicates, our method achieves competitive performance while being more than 74× more compact than LangSplat and 245× more compact than Feature-3DGS. Particularly, when CLIP features are extracted for each segment using SAM masks, each region is represented by a single feature vector. In this case, our adaptive sparsification enables effective merging, allowing the 3D feature field to be described with only 5% of the Gaussians compared to existing methods.

The following experiment addresses a more general scenario than the previous two feature maps. LSeg, based on the DPT [37] backbone, and CLIP with SAM both produce features at nearly the same resolution as the input image. In contrast, MaskCLIP produces low-resolution, patch-level features, which lead to performance degradation in the baseline. Our approach compensates for the limitations of these coarse features by using high-resolution reference features during adaptive sparsification. As shown in Tab. 4 and Fig. 6, our method provides a representation over 182× more compact than Feature-3DGS, while achieving more than 30% mIoU improvement and effectively removing noisy activations.

5.3. Ablation

We conducted an ablation study in Tab. 3 to demonstrate the effectiveness of each component of our pipeline. Ablations were performed for all experiments presented in Sec. 5.2. In particular, a key component of our method is the adaptive sparsification (Sec. 4.3) that eliminates redundant Gaussians. The merging step contributes to an additional 70% storage reduction. In addition, variance filtering (Sec. 4.1) effectively removes noisy features from low-resolution features from MaskCLIP, contributing to improved performance. After the feature compression stage, the number of Gaussians remains unchanged, but compressing high-dimensional features into a low-dimensional space contributes significantly to storage reduction.

5.4. Open-vocabulary 3D Segmentation

We additionally perform open-vocabulary 3D segmentation by directly querying the features embedded in the Gaussians. To associate CF^3 with pre-trained 3DGS, we establish a

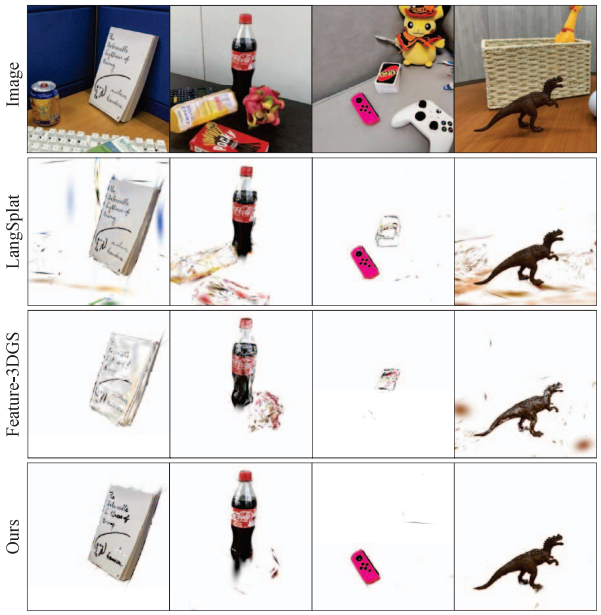

Figure 7. **3D Segmentation Results.** We perform open-vocabulary 3D segmentation on the 3D-OVS dataset. The following queries are used, in order: *a book of The Unbearable Lightness of Being, Coca-Cola, a red Nintendo Switch Joy-Con controller, and Dinosaur.*

	Storage↓	FPS↑	mIoU↑	#G↓
Feature-3DGS(128)	305.5M	90.7	81.4	421k
LangSplat	83.9M	27.7	81.9	332k
3DGS*	746.8M	2.6	82.8	332k
CF^3 (Ours)	**1.7M**	**140.3**	**84.5**	**21k**

Table 5. **Results on 3D-OVS Dataset with CLIP+SAM [20, 36].**

mapping from CF^3 to the pre-trained 3DGS after applying feature lifting (Sec. 4.1). Each feature-lifted 3DGS point is mapped to its closest CF^3 point by identifying the k=3 nearest neighbors in coordinate space and selecting the one with the highest cosine similarity in feature space. This allows us to propagate the text-based query results from CF^3 back to the 3DGS for visualization.

We perform 3D segmentation on the 3D-OVS dataset [28]. Specifically, the evaluation is conducted on the Office desk, Room, Snacks, and Sofa scenes included in the dataset. Unlike LangSplat [35] and Feature-3DGS [56], which train the autoencoder or decoder in 2D before lifting, our method learns the autoencoder directly on lifted 3D features, preserving the feature distribution between training and inference. As shown in Fig. 7, this leads to improved 3D segmentation performance. Open-vocabulary 2D segmentation results on the same dataset are also reported in Tab. 5.

To demonstrate the efficiency of our feature field representation, we conduct experiments on the large-scale outdoor KITTI-360 dataset [27]. Large-scale scenes pose a significant challenge for traditional optimization-based feature

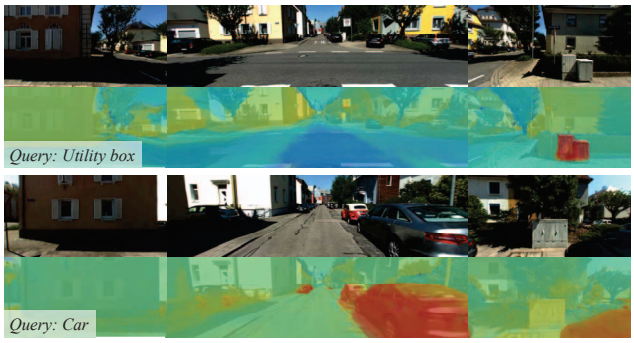

Figure 8. **Additional Result on KITTI-360 Dataset.** We visualize each Gaussian in CF^3 based on its similarity to the text query and render the result. Blue indicates low similarity, while higher similarity is shown in red.

	Storage↓	FPS↑	#G↓
3DGS*	3810.2M	1.8	1734k
CF^3 (Ours)	**6.2M**	**141.6**	**95k**

Table 6. **Results on KITTI-360 Dataset.**

embedding due to their high computational cost. As shown in Tab. 6, by leveraging a highly compact representation, CF^3 substantially reduces storage overhead while enabling real-time rendering speeds. Fig. 8 shows a visualization of the feature similarity between each Gaussian and a given text query. We compute the similarity directly between the embedded feature in each Gaussian and the text query feature, and map this similarity to a color for visualization. Importantly, this is based purely on the 3D Gaussian features, not on rendered features in 2D. These results highlight the potential of CF^3 for open-vocabulary semantic segmentation and localization in large-scale environments.

6. Conclusion

This paper presents a pipeline for constructing compact and fast 3D feature fields (CF^3). Unlike prior approaches, we train a per-Gaussian autoencoder on features lifted via weighted multi-view fusion. In addition, we propose an adaptive sparsification strategy that prunes and merges redundant Gaussians, reducing their count while maintaining representation fidelity. Unlike other 3D feature field compression methods that store high-dimensional attributes separately and rely on auxiliary data structures such as hash grids, our method stores 3D features directly in the RGB channels of 3DGS, replacing color with features. This design makes it compatible with existing 3DGS pipelines. While feature lifting is fast and efficient, the overall pipeline currently takes approximately 30 minutes per scene due to the autoencoder training and sparsification stages. We plan to accelerate these stages to minimize the overhead.

Acknowledgements.

This work was supported by IITP grant (RS-2021-II211343: AI Graduate School Program at Seoul National Univ. (5%) and RS-2023-00227993: Detailed 3D reconstruction for urban areas from unstructured images (60%)) and NRF grant (No.2023R1A1C200781211 (35%)) funded by the Korea government (MSIT).

References

[1] Milena T Bagdasarian, Paul Knoll, Yi-Hsin Li, Florian Barthel, Anna Hilsmann, Peter Eisert, and Wieland Morgenstern. 3dgs. zip: A survey on 3d gaussian splatting compression methods. *arXiv preprint arXiv:2407.09510*, 2024. 4

[2] M. T. Bagdasarian, P. Knoll, Y. Li, F. Barthel, A. Hilsmann, P. Eisert, and W. Morgenstern. 3DGS.zip: A Survey on 3D Gaussian Splatting Compression Methods. *Computer Graphics Forum*, page e70078, 2025. https://w-m.github.io/3dgs-compression-survey/. 4

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5470–5479, 2022. 4

[4] Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Hac: Hash-grid assisted context for 3d gaussian splatting compression. In *European Conference on Computer Vision*, pages 422–438. Springer, 2024. 3, 4

[5] Yue Chen, Xingyu Chen, Anpei Chen, Gerard Pons-Moll, and Yuliang Xiu. Feat2gs: Probing visual foundation models with gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 6348–6361, 2025. 2

[6] Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Hac++: Towards 100x compression of 3d gaussian splatting. *arXiv preprint arXiv:2501.12255*, 2025. 4

[7] Jiahuan Cheng, Jan-Nico Zaech, Luc Van Gool, and Danda Pani Paudel. Occam's lgs: A simple approach for language gaussian splatting. *arXiv preprint arXiv:2412.01807*, 2024. 3, 4

[8] Kai Cheng, Xiaoxiao Long, Kaizhi Yang, Yao Yao, Wei Yin, Yuexin Ma, Wenping Wang, and Xuejin Chen. Gaussianpro: 3d gaussian splatting with progressive propagation. In *Forty-first International Conference on Machine Learning*, 2024. 3

[9] Mohamed El Banani, Amit Raj, Kevis-Kokitsi Maninis, Abhishek Kar, Yuanzhen Li, Michael Rubinstein, Deqing Sun, Leonidas Guibas, Justin Johnson, and Varun Jampani. Probing the 3d awareness of visual foundation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21795–21806, 2024. 2

[10] Francis Engelmann, Fabian Manhardt, Michael Niemeyer, Keisuke Tateno, Marc Pollefeys, and Federico Tombari. Open-set 3d scene segmentation with rendered novel views. 2023. 3

[11] Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. *arXiv preprint arXiv:2311.17245*, 2023. 3, 4, 5, 6

[12] Zhiwen Fan, Jian Zhang, Wenyan Cong, Peihao Wang, Renjie Li, Kairun Wen, Shijie Zhou, Achuta Kadambi, Zhangyang Wang, Danfei Xu, et al. Large spatial model: End-to-end unposed images to semantic 3d. *Advances in Neural Information Processing Systems*, 37:40212–40229, 2025. 3

[13] Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of gaussians. In *European Conference on Computer Vision*, pages 165–181. Springer, 2024. 3

[14] Xiao Fu, Shangzhan Zhang, Tianrun Chen, Yichong Lu, Lanyun Zhu, Xiaowei Zhou, Andreas Geiger, and Yiyi Liao. Panoptic nerf: 3d-to-2d label transfer for panoptic urban scene segmentation. In *2022 International Conference on 3D Vision (3DV)*, pages 1–11. IEEE, 2022. 3

[15] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Scaling open-vocabulary image segmentation with image-level labels. In *European Conference on Computer Vision*, pages 540–557. Springer, 2022. 3

[16] Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. Eagles: Efficient accelerated 3d gaussians with lightweight encodings. In *European Conference on Computer Vision*, pages 54–71. Springer, 2024. 3

[17] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 3, 5

[18] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19729–19739, 2023. 1, 3, 5, 6

[19] Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Jeff Tseng, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 3d gaussian splatting as markov chain monte carlo. *arXiv preprint arXiv:2404.09591*, 2024. 3

[20] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 1, 6, 8

[21] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing nerf for editing via feature field distillation. *Advances in Neural Information Processing Systems*, 35:23311–23330, 2022. 3

[22] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12871–12881, 2022. 3

[23] 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*, pages 21719–21728, 2024. 3

[24] Soonbin Lee, Fangwen Shu, Yago Sanchez, Thomas Schierl, and Cornelius Hellge. Compression of 3d gaussian splatting with optimized feature planes and standard video codecs. *arXiv preprint arXiv:2501.03399*, 2025. 4

[25] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and René Ranftl. Language-driven semantic segmentation. *arXiv preprint arXiv:2201.03546*, 2022. 1, 3, 5, 6

[26] Siyun Liang, Sen Wang, Kunyi Li, Michael Niemeyer, Stefano Gasperini, Nassir Navab, and Federico Tombari. Supergseg: Open-vocabulary 3d segmentation with structured super-gaussians. *arXiv preprint arXiv:2412.10231*, 2024. 2, 3

[27] Yiyi Liao, Jun Xie, and Andreas Geiger. Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3292–3310, 2022. 8

[28] Kunhao Liu, Fangneng Zhan, Jiahui Zhang, Muyu Xu, Yingchen Yu, Abdulmotaleb El Saddik, Christian Theobalt, Eric Xing, and Shijian Lu. Weakly supervised 3d open-vocabulary segmentation. *Advances in Neural Information Processing Systems*, 36:53433–53456, 2023. 3, 8

[29] 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*, pages 20654–20664, 2024. 3

[30] Juliette Marrie, Romain Ménégaux, Michael Arbel, Diane Larlus, and Julien Mairal. Ludvig: Learning-free uplifting of 2d visual features to gaussian splatting scenes. *arXiv preprint arXiv:2410.14462*, 2024. 3, 4

[31] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 1

[32] Wieland Morgenstern, Florian Barthel, Anna Hilsmann, and Peter Eisert. Compact 3d scene representation via self-organizing gaussian grids. In *European Conference on Computer Vision*, pages 18–34. Springer, 2024. 3

[33] K Navaneet, Kossar Pourahmadi Meibodi, Soroush Abbasi Koohpayegani, and Hamed Pirsiavash. Compact3d: Compressing gaussian splat radiance field models with vector quantization. *arXiv preprint arXiv:2311.18159*, 2(3), 2023. 4

[34] 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*, pages 10349–10358, 2024. 3

[35] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20051–20060, 2024. 1, 2, 3, 4, 6, 8

[36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1, 6, 8

[37] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 7

[38] 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. *arXiv preprint arXiv:2403.17898*, 2024. 3

[39] Dennis Schieferdecker and Marco F Huber. Gaussian mixture reduction via clustering. In *2009 12th international conference on information fusion*, pages 1536–1543. IEEE, 2009. 5

[40] Jin-Chuan Shi, Miao Wang, Hao-Bin Duan, and Shao-Hua Guan. Language embedded 3d gaussians for open-vocabulary scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5333–5343, 2024. 2, 3

[41] Seungjoo Shin, Jaesik Park, and Sunghyun Cho. Locality-aware gaussian compression for fast and high-quality rendering. In *Proceedings of the Int. Conf. on Learning Representations (ICLR)*, 2025. 3

[42] Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Buló, Norman Müller, Matthias Nießner, Angela Dai, and Peter Kontschieder. Panoptic lifting for 3d scene understanding with neural fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9043–9052, 2023. 3

[43] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 5

[44] Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. Neural feature fusion fields: 3d distillation of self-supervised 2d image representations. In *2022 International Conference on 3D Vision (3DV)*, pages 443–453. IEEE, 2022. 3

[45] Bing Wang, Lu Chen, and Bo Yang. Dm-nerf: 3d scene geometry decomposition and manipulation from 2d images. *arXiv preprint arXiv:2208.07227*, 2022. 3

[46] Henan Wang, Hanxin Zhu, Tianyu He, Runsen Feng, Jiajun Deng, Jiang Bian, and Zhibo Chen. End-to-end rate-distortion optimized 3d gaussian representation. In *European Conference on Computer Vision*, pages 76–92. Springer, 2024. 3

[47] Xingrui Wang, Cuiling Lan, Hanxin Zhu, Zhibo Chen, and Yan Lu. Gsemsplat: Generalizable semantic 3d gaussian splatting from uncalibrated image pairs. *arXiv preprint arXiv:2412.16932*, 2024. 3

[48] Yanmin Wu, Jiarui Meng, Haijie Li, Chenming Wu, Yahao Shi, Xinhua Cheng, Chen Zhao, Haocheng Feng, Errui Ding, Jingdong Wang, et al. Opengaussian: Towards point-level

3d gaussian-based open vocabulary understanding. *arXiv preprint arXiv:2406.02058*, 2024. 2, 3

[49] Shuzhao Xie, Weixiang Zhang, Chen Tang, Yunpeng Bai, Rongwei Lu, Shijia Ge, and Zhi Wang. Mesongs: Post-training compression of 3d gaussians via efficient attribute transformation. In *European Conference on Computer Vision*, pages 434–452. Springer, 2024. 3

[50] Yuanwen Yue, Anurag Das, Francis Engelmann, Siyu Tang, and Jan Eric Lenssen. Improving 2D Feature Representations by 3D-Aware Fine-Tuning. In *European Conference on Computer Vision (ECCV)*, 2024. 2, 4

[51] Xiaoshuai Zhang, Abhijit Kundu, Thomas Funkhouser, Leonidas Guibas, Hao Su, and Kyle Genova. Nerflets: Local radiance fields for efficient structure-aware 3d scene representation from 2d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8274–8284, 2023. 3

[52] Xiaoshuai Zhang, Zhicheng Wang, Howard Zhou, Soham Ghosh, Danushen Gnanapragasam, Varun Jampani, Hao Su, and Leonidas Guibas. Condense: Consistent 2d/3d pre-training for dense and sparse features from multi-view images. In *European Conference on Computer Vision*, pages 19–38. Springer, 2024. 2

[53] Zheng Zhang, Wenbo Hu, Yixing Lao, Tong He, and Hengshuang Zhao. Pixel-gs: Density control with pixel-aware gradient for 3d gaussian splatting. In *European Conference on Computer Vision*, pages 326–342. Springer, 2024. 3

[54] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15838–15847, 2021. 3

[55] Chong Zhou, Chen Change Loy, and Bo Dai. Extract free dense labels from clip. In *European Conference on Computer Vision*, pages 696–712. Springer, 2022. 5, 7

[56] Shijie Zhou, Haoran Chang, Sicheng Jiang, Zhiwen Fan, Zehao Zhu, Dejia Xu, Pradyumna Chari, Suya You, Zhangyang Wang, and Achuta Kadambi. Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21676–21685, 2024. 1, 2, 3, 5, 6, 7, 8

[57] Xingxing Zuo, Pouya Samangouei, Yunwen Zhou, Yan Di, and Mingyang Li. Fmgs: Foundation model embedded 3d gaussian splatting for holistic 3d scene understanding. *International Journal of Computer Vision*, pages 1–17, 2024. 2, 3