Occam's LGS: An efficient approach for Language Gaussian Splatting

Jiahuan Cheng[1,2]
jcheng65@jh.edu
Jan-Nico Zaech[2]
jan-nico.zaech@insait.ai
Luc Van Gool[2]
luc.vangool@insait.ai
Danda Pani Paudel[2]
danda.paudel@insait.ai

[1] Johns Hopkins University
Baltimore, United States
[2] INSAIT
Sofia University
Sofia, Bulgaria

Abstract

Gaussian Splatting is a widely adopted approach for 3D scene representation, offering efficient, high-quality reconstruction and rendering. A key reason for its success is the simplicity of representing scenes with sets of Gaussians, making it interpretable and adaptable. To enhance understanding beyond visual representation, recent approaches extend Gaussian Splatting with semantic vision-language features, enabling open-set tasks. Typically, these language features are aggregated from multiple 2D views, however, existing methods rely on cumbersome techniques, resulting in high computational costs and longer training times. In this work, we show that the complicated pipelines for language 3D Gaussian Splatting are simply unnecessary. Instead, we follow a probabilistic formulation of Language Gaussian Splatting and apply Occam's razor to the task at hand, leading to a highly efficient weighted multi-view feature aggregation technique. Doing so offers us state-of-the-art results with a speed-up of two orders of magnitude without any compression, allowing for easy scene manipulation.

1 Introduction

3D semantic understanding enables machines to interpret and reason about the physical world, with applications spanning robotics [26], autonomous navigation [32, 41], and augmented reality. While traditional approaches focused on extracting geometric structures using predefined categories [9], recent advances have integrated natural language capabilities. This emerging field of 3D language understanding [4, 7, 10, 13, 17, 22, 31, 40] connects linguistic expressions with 3D visual representations (point clouds, meshes, RGB-D images), enabling more flexible and comprehensive scene interpretation.

Recent breakthroughs in 3D scene representation [12, 18, 30, 38], particularly 3D Gaussian Splatting [12, 39], have revolutionized scene reconstruction and rendering by offering an efficient and high-quality approach to modeling 3D environments. While 3D Gaussian Splatting excels at geometric reconstruction and novel view synthesis, its potential for semantic

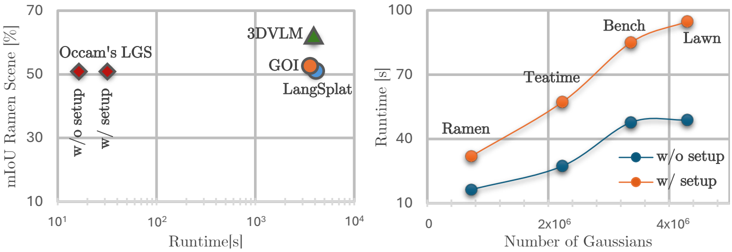

Figure 1: Occam's LGS performs training-free language feature aggregation in 3D by accurately modeling rendering of Gaussian Splatting, avoiding expensive training in feature space. This improves runtime by two orders of magnitude in comparison with 3DVLM [20], GOI [24], and LangSplat [23], while achieving SOTA performance for open-set vision-language tasks and allowing for downstream applications such as scene editing.

understanding remains largely unexplored. Recent research has started to address this limitation by incorporating language features into the Gaussian representations and taken first steps towards open-vocabulary scene understanding through natural language queries.

Vision-language models like CLIP [2] and LSeg [5] have demonstrated the power of joint visual-linguistic understanding by training on vast image-text pairs, while foundation models like Segment Anything (SAM) [14] and DINO-V2 [19] have shown the potential of large-scale visual learning. Although these models have changed 2D image understanding completely, extending their capabilities to 3D domains remains challenging [1, 20, 23, 29, 35, 36]: directly learning 3D-language representations lacks paired data and 2D vision-language models typically produce expressive high-dimensional features, which poses an often infeasible computational cost when integrated in complex 3D scenes.

Current approaches address this challenge through scene-specific feature autoencoders [23] or quantization to create a compact 3D feature space [29]. However, these scene-specific compression approaches fundamentally limit scalability and generalization. While effective for individual environments, they create barriers for multi-scene applications by requiring separate training for each scene, increasing both computational and storage overhead. Moreover, these scene-specific solutions make it difficult to transfer knowledge between environments and limit the potential for open-world understanding.

We take a fundamentally different approach by proposing a training-free method, that avoids repeated iteration over views, as well as loss computation in the high-dimensional feature space. Overall, our contribution can be summarized as follows:

- A training-free global optimization approach for Language 3D Gaussian Splatting, improving speed by two orders of magnitude.
- A reasoning by rendering approach to avoiding expensive feature rendering by following a probabilistic formulation of the Gaussian Splatting forward process.
- Compatibility with uncompressed and arbitrary dimensional language features.

2 Related works

3D Open Vocabulary Scene Understanding. Recent works [3, 6, 22, 33, 40] have explored 2D vision-language models such as CLIP [25], OpenSeg [5], and OVSeg [16], and foundation models such as SAM [14] to ground 2D language-aligned features in 3D space. Initial approaches [11, 13, 15] demonstrated the feasibility of distilling language features into NeRF-based representations, though at the cost of slow training and rendering. More recent Gaussian-based methods have improved efficiency while following a common framework: extracting language-aligned feature maps from 2D views and grounding them to 3D space through multi-view optimization. LangSplat [23] employs SAM to obtain masks at different granularity, then extracts CLIP image features for these regions. LEGaussians [29] combines DINO [2] and CLIP features to obtain hybrid dense language feature maps and FMGS [43] uses a similar combination to improve robustness. GOI [24] leverages APE [28] to extract pixel-aligned features with fine boundaries.

However, incorporating the high-dimensional language features into 3D space presents unique challenges. It not only increases memory requirements but also significantly extends training time. Different approaches have been proposed to address this challenge: LangSplat [23] employs a scene-specific autoencoder for feature compression, LEGaussians [29] uses quantization of semantic features, and Feature-3DGS [42] employs feature compression with a lightweight decoder. FMGS [43] introduces multi-resolution encoding for efficient semantic embedding and GOI [24] utilizes a trainable feature clustering codebook. In contrast to these methods, our approach supports arbitrary-dimensional uncompressed features without compromising training speed. Our method preserves semantic meaning without requiring additional preprocessing steps or encoder-decoder structures. This ensures open-vocabulary generalization to the same extent as the underlying feature extractor and enables downstream applications using dynamic scenes or scene editing.

Uplifting 2D Features to 3D. While current methods [15, 20, 23, 29] focus on optimization-based feature lifting, several approaches also explore direct back-projection of 2D features to 3D space using pixel rays and depth information. OpenGaussian [35] proposes an instance-level 3D-2D feature association method that links 3D points to 2D masks. Semantic Gaussians [8] utilizes depth to find corresponding 3D Gaussians along each pixel's ray. PLGS [34] initializes semantic anchor points by back-projecting pixels into 3D space using camera parameters. While not focusing on semantic understanding, FlashSplat [27] demonstrates that 2D mask rendering is essentially a linear function of Gaussian labels, enabling closed-form optimization through linear programming. In contrast, our method accurately models Gaussian Splatting forward rendering to lift 2D language features to 3D, while avoiding back-projection limitations such as occlusion and incomplete depth information.

3 Method

The following section provides an overview of our method: 1) introduction of Gaussian splatting rendering, 2) our formulation using maximum likelihood features uplifting, and 3) its implementation via forward rendering and postprocessing by filtering. An overview of the flow of our method is visually presented in Figure 2.

3.1 Gaussian Splatting Rendering

3D Gaussian Splatting [12] represents a scene as a collection of anisotropic Gaussians, where each Gaussian is defined by its position $\mathbf{x} \in \mathbb{R}^3$, covariance $\Sigma \in \mathbb{R}^6$, color represented by either a 3D color representation $\mathbf{c} \in \mathbb{R}^3$, or a viewpoint dependent representation as spherical harmonics (SH) parameters, and opacity $o \in \mathbb{R}$: $G(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x})^\intercal \Sigma^{-1}(\mathbf{x})}$. The primitive representation of a Gaussian is defined as $G = \{\mathbf{x}, \Sigma, \mathbf{c}, o\}$, and let $\mathcal{G} = \{G_i\}_{i=1}^N$ be the set of

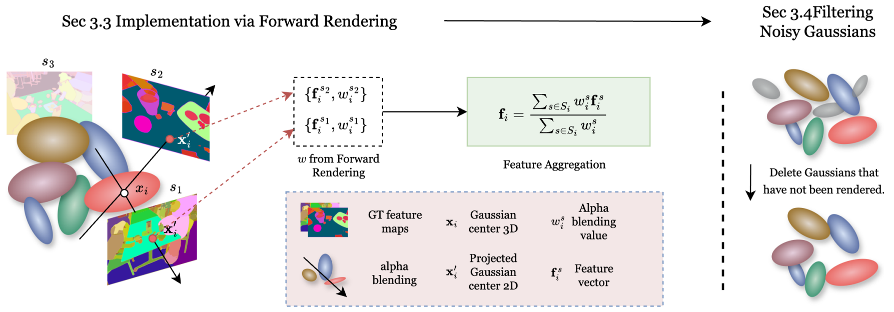

Figure 2: Overview of our method: Occam's LGS consists of two main stages: (1) Forward rendering process of 3D Gaussian Splatting to obtain alpha blending weights w, projected positions x'_i for each Gaussian and their corresponding pixels p_i in 2D views, followed by weighted aggregation of multi-view semantic features (Sec. 3.3); and (2) Filtering of noisy Gaussians that remain invisible throughout the rendering process (Sec. 3.4).

Gaussians in a scene. For rendering, we first project each 3D Gaussian onto the image plane, resulting in a 2D Gaussian $G' = \{\mathbf{x}', \Sigma', \mathbf{c}, o\}$, with $\mathbf{x}' \in \mathbb{R}^2$ as the projected 2D coordinates and $\Sigma' \in \mathbb{R}^{2\times 2}$ as the projected covariance matrix.

To render the color at pixel $\mathbf{p}$, we blend the contributions of all Gaussians along the ray originating from the camera center, say $\mathcal{G}^r = \{G_k\}_{k=1}^R$, where R is the total number of Gaussians along the ray. To achieve efficient rendering in the Gaussian Splatting framework, this rendering process is implemented through a tile-based sorting strategy, which utilizes the approximation of Gaussians by a spatially truncated distribution. For this purpose, the image is partitioned into tiles, typically spanning 16x16 pixels. Each 2D Gaussian is assigned to all tiles intersecting with a bounding box placed at the 3σ boundary. Within each tile, Gaussians are sorted by depth from the camera center. These sorted Gaussians represent the collection of Gaussians that potentially contribute to any pixel within that tile, which are then rendered sequentially, starting closest to the camera.

For the i-th Gaussian G_i in the sorted list, its contribution to a pixel $\mathbf{p}$ is defined as $\alpha_i = \mathbf{o}_i \cdot e^{-\frac{1}{2}(\mathbf{p}-\mathbf{x}'_i)^\intercal \Sigma^{-1}(\mathbf{p}-\mathbf{x}'_i)}$, representing its influence. When projecting all relevant Gaussians, the color of the pixel $\mathbf{p}$ is computed through a weighted blending process given by

$$\mathcal{C}(\mathbf{p}) = \psi_c(\mathbf{p}, \mathcal{G}) = \sum_{i=1}^{R} \alpha_i \mathbf{c}_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{1}$$

where, $\psi_c(,.,)$ is the color rendering function.

With the goal of lifting multi-view 2D features into 3D space, we extend each Gaussian representation to include a high-dimensional semantic feature vector $\mathbf{f} \in \mathbb{R}^d$. Thus, each augmented 3D Gaussian G_i learns its own semantic feature $\mathbf{f_i}$. Similar to color blending, the semantic feature at pixel $\mathbf{p}$ is computed through alpha-weighted blending

$$\mathcal{F}(\mathbf{p}) = \psi_f(\mathbf{p}, \mathcal{G}) = \sum_{i=1}^{R} w_i \mathbf{f}_i \tag{2}$$

where ψ_f is the feature rendering function and $w_i = \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j)$ represents the alpha blending weight for Gaussian i.

It is important to note that explicit rendering of semantics according to Equation (2) is expensive both in memory and computation when compared to the rendering of color in Equation (1). This is due to the considerably higher dimensionality of the features $d >> 3$, which prevents the GPU from fitting all required data in the shader L1 cache [23].

3.2 Maximum Likelihood Feature Uplifting

We deeply ground our approach to feature uplifting in the principles of 3DGS and probabilistic modeling. While conventional semantic feature learning methods [8, 35] rely on back-projection, where 2D pixel-space language features are projected back to 3D space through ray-tracing, we leverage the underlying mathematical structure of Gaussian Splatting's forward rendering process to establish a more precise and theoretically sound feature lifting mechanism and relate it to forward modeling.

We formulate the feature uplifting problem as a Maximum-a-Posteriori (MAP) estimation problem. Given 2D feature maps $\mathbf{F}^{2D}$, the MAP estimator of 3D features $\mathbf{f}$ is:

$$\arg\max_{\mathbf{f}} p(\mathbf{f}|\mathbf{F}^{2D}) = \arg\max_{\mathbf{f}} p(\mathbf{F}^{2D}|\mathbf{f}) \frac{p(\mathbf{f})}{p(\mathbf{F}^{2D})} \tag{3}$$

For language gaussian splatting with uniformly distributed language features $p(\mathbf{f})$, which we assume for the efficient feature space of CLIP, the estimator simplifies to the Maximum Likelihood (ML) formulation $\arg\max_{\mathbf{f}} p(\mathbf{F}^{2D}|\mathbf{f})$.

Under a Gaussian noise model $\varepsilon_{p,v}$ for language features and conditional independence of pixels, the rendered 2D feature at pixel p in view v is $\mathbf{f}^{2D}_{p,v} = \sum_{k=1}^{R} w_{k,p,v}\mathbf{f}_k + \varepsilon_{p,v}$. This leads to the weighted aggregation formula for the 3D feature of each Gaussian

$$\boxed{\mathbf{f}_i = \frac{\sum\limits_{s \in S_i} w_i^s \mathbf{f}_i^s}{\sum\limits_{s \in S_i} w_i^s}} \tag{4}$$

where S_i is the set of views where Gaussian i is visible, w_i^s represents the contribution (alpha blending weight) of Gaussian i at its center projection in view s, and $\mathbf{f}_i^s$ is the 2D feature at Gaussian i's center projected pixel in view s. This formula emerges naturally from our probabilistic framework and provides a solution that weights each view's contribution according to the Gaussian's influence in that view. We simplify the framework to use only the single most influential pixel's feature rather than aggregating across all affected pixels for computational speed and efficiency. This simplification is reasonable given that semantic features exhibit spatial coherence—nearby pixels typically share similar semantic representations. Despite this simplification, our experiments in Sec 4 demonstrate that this approach maintains strong performance across all benchmarks. Views, where a Gaussian is more prominent (with a higher alpha blending weight), contribute more significantly to its 3D feature, addressing the challenge of noisy or partially occluded perspectives. We provide complete step-by-step derivation in the supplementary material.

3.3 Implementation via Forward Rendering

Our weighted aggregation formula directly maps to the forward rendering process, enabling an efficient implementation that avoids complex back-projection or iterative optimization.

Figure 3: Qualitative comparison of relevancy score visualization.

Given a trained visual Gaussian model, for each Gaussian G_i we perform the following steps: **1)** Identify the set of views S_i where the Gaussian G_i is visible (within the view frustum). **2)** Determine the Gaussian's center projected pixel location $\mathbf{p}_i^s = \lfloor \mathbf{x}_i' \rfloor$ in each view s. **3)** Record the alpha blending weight w_i^s and extract the 2D feature $\mathbf{f}_i^s$ at each center projected pixel. **4)** Compute the weighted average following Equation 4.

This implementation leverages the existing tile-based rendering architecture, allowing us to record projection locations and alpha values during the optimized standard rendering pass. The forward feature collection approach offers several key advantages; First, the feature uplifting requires only a single forward pass through the views, making it computationally efficient. Second, view frustum culling and tile-based sorting naturally handle occluded Gaussians, ensuring we only process visible features. Finally, by explicitly recording projection locations during rendering, we establish direct and precise correspondence between 2D features and 3D Gaussians, eliminating the need for iterative optimization methods.

3.4 Filtering Noisy Gaussians

After the forward rendering pass, we identify and filter out Gaussians that have no contributions across all views. This filtering effectively reduces noise while significantly reducing the number of Gaussians. This statistical outlier elimination is consistent with our probabilistic framework, as Gaussians with minimal visibility have high uncertainty in their feature estimates. Through this process, we obtain a refined set of semantic Gaussians $\mathcal{G}^s = \{G_i^s\}_{i=1}^M$, where $M \ll N$, representing only the Gaussians that meaningfully contribute to the scene's semantic representation.

4 Experiments

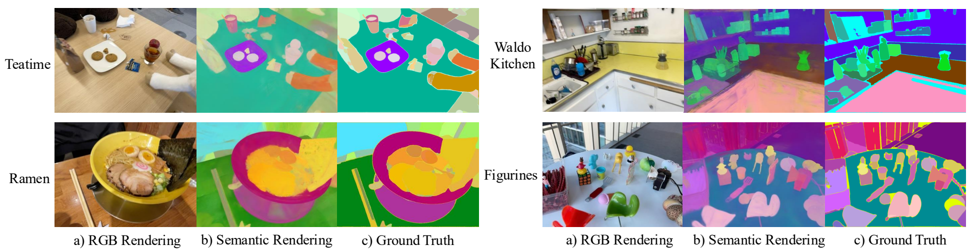

Figure 4: PCA visualization of semantic features on LERF Dataset. (a) Original RGB rendering. (b) PCA visualization of our rendered semantic feature maps. (c) Initial semantic feature maps extracted using CLIP from SAM segmentations.

4.1 Dataset

To assess the effectiveness of our method, we evaluate it on two widely used datasets: LERF [13] and 3D-OVS [17]. The LERF dataset features everyday indoor scenes with diverse objects and we perform quantitative evaluations on the commonly used scenes Ramen, Figurines, Teatime, and Waldo Kitchen according to the LERF evaluation protocol [13]. For these scenes, we evaluate both open-vocabulary segmentation (mIoU) and localization accuracy. The 3D-OVS dataset contains long-tail objects in diverse backgrounds, and we evaluate segmentation performance on five scenes: sofa, lawn, bed, room, and bench. For evaluation, we utilize the annotations provided by [23] for both datasets. On this data, we further show object insertion as a downstream application enabled by our method.

4.2 Implementation Details

Our implementation of Occam's LGS is built on gsplat [37], a CUDA-accelerated Gaussian rendering library that supports high-dimensional feature rendering. For each scene, we train a 3D Gaussian Splatt for 30,000 iterations using the common Gaussian Splatting [12] parameters. To ensure a fair comparison, we follow the 2D language feature map generation method from [23], leveraging OpenCLIP ViT-B/16 [25] and SAM ViT-H [14]. Following the feature extraction, we use our method to uplift the 2D language features to 3D scenes using training views only, and evaluate the results on separate testing views, withheld during training of the semantic representation. More details can be found in the supplementary.

For the LERF dataset, we follow the LERF [13] evaluation protocol, computing relevancy scores between query text and rendered 2D feature maps with a mask threshold of 0.5. For 3D-OVS, we first normalize the rendered feature maps to ensure comparable relevancy scores across classes, then apply dynamic thresholding detailed in the supplementary.

4.3 Open Vocabulary Semantic Segmentation

Open vocabulary semantic segmentation is the task to identify and mask all regions in a 3D scene that match a given language query, without being limited to predefined categories. We evaluate our approach against existing methods on two datasets: LERF and 3D-OVS, with quantitative results shown in Table 1 respectively and qualitative results shown in Figure 3 and 4. On the LERF dataset, under identical experimental settings as LangSplat, our approach demonstrates a substantial improvement of 9.9% mIoU. Notably, our method shows advantages, especially in complex scenes with sparse view coverage. An example of this is Waldo Kitchen, where the camera captures an entire room rather than focusing on a specific area that is fully surrounded by camera views.

On 3D-OVS, our method achieves state-of-the-art results, with an improvement of 1.6% mIoU over the current SOTA LangSplat [23]. Performance gains of 3D-OVS are less pronounced compared to LERF [13], likely since this dataset contains simpler scenes, focused on a single region with good view coverage and fewer objects, where existing methods have already saturated in performance.

4.4 Localization

We further evaluate on the language-guided localization task, where, given a language query, the pixel with the highest relevancy score is selected, following [13]. Benchmarking against

Method	LERF Dataset					3D-OVS Dataset					
	Ramen	Figurines	Teatime	Waldo Kitchen	Overall	Bed	Bench	Room	Sofa	Lawn	Overall
LSeg	7.0	7.6	21.7	29.9	16.6	56.0	6.0	19.2	4.5	17.5	20.6
LERF	28.2	38.6	45.0	37.9	37.4	73.5	53.2	46.6	27.0	73.7	54.8
GS-Grouping	45.5	60.9	40.0	38.7	46.3	83.0	91.5	85.9	87.3	90.6	87.7
Feature-3DGS	43.7	58.8	40.5	39.6	45.7	83.5	90.7	84.7	86.9	93.4	87.8
LEGaussians	46.0	60.3	40.8	39.4	46.9	84.9	91.1	86.0	87.8	92.5	88.5
GOI	**52.6**	**63.7**	44.5	41.4	50.6	89.4	92.8	91.3	85.6	94.1	90.6
LangSplat	51.2	44.7	65.1	44.5	51.4	92.5	94.2	94.1*	**90.0**	96.1	93.4
Ours	51.0	58.6	**70.2**	**65.3**	**61.3**	**96.8**	**95.8**	**96.5***	88.8	**97.0**	**95.0**

Table 1: Comparison of average IoU on LERF and 3D-OVS Datasets. For the 3D-OVS Dataset, the utilized image resolution is 1440 × 1080. Methods with * are evaluated on testing-data that has an annotation error in the 3D-OVS testset corrected.

Method	Ramen	Figurines	Teatime	Waldo
Language Segmentation	14.1	8.9	33.9	27.3
LERF	62.0	75.0	84.8	72.7
GS-Grouping	68.6	75.0	74.3	88.2
Feature-3DGS	69.8	77.2	73.4	87.6
Language Gaussians	67.5	75.6	75.2	90.3
GOI	75.5	**88.6**	82.9	90.4
Language Splatting	73.2	80.4	88.1	**95.5**
Our Method	**74.7**	80.4	**93.2**	81.8

Table 2: Comparison of localization accuracy on LERF Datasets.

Feature Dimensions	Ramen	Figurines	Teatime	Waldo
16-Dim	48.4	57.7	75.9	59.9
32-Dim	**53.6**	61.8	78.6	62.6
64-Dim	51.7	**63.4**	**79.0**	63.7
128-Dim	52.1	62.4	74.3	62.6
256-Dim	53.0	62.6	74.3	61.1
512-Dim	51.0	58.6	70.2	**65.3**

Table 3: Average mean Intersection over Union (mIoU) with varying feature dimensions.

existing methods, our approach achieves competitive localization accuracy, as shown by Table 2. Adding feature compression, as in existing methods, increases the localization accuracy of our approach to 84.1. This is due to the metric only measuring the closest matching pixel while ignoring the feature distribution and object shape, making it prone to noise. We provide detailed experiments in this setting in the supplementary material.

4.5 Robustness to Feature Dimension Variations

In the following, we demonstrate the advantages of different feature dimensions that Occam's LGS enables and assess the respective scalability and performance. For these experiments, we 1) used an autoencoder to compress the CLIP features, 2) uplifted them to 3D using our method, 3) rendered the compressed features, and 4) decompressed the features to 512D using the autoencoder. Table 3 shows the IoU across different feature dimensions on LERF datasets. Notably, segmentation performance improves to 64.5 %mIoU with 64D features, outperforming the LangSplat [23] baseline by 13.1% mIoU. This confirms that feature compression filters noise by limiting the feature space to objects present in the 3d scene, at the cost of limited scene editing capability. Since our method supports arbitrary feature dimensions, selecting this operating point is straightforward and can be used depending on the task at hand.

4.6 Computational Efficiency

We evaluate our method's computational efficiency against existing approaches in Figure 1. While LangSplat [23], GOI [24], and 3DVLM [21] require 60-67 minutes to uplift three levels of features, our approach completes the same task in just 16.4-31.9 seconds. This

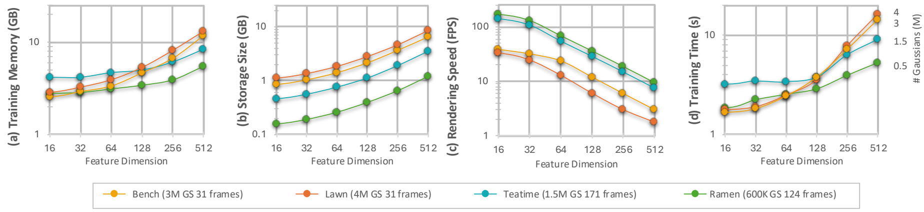

Figure 5: Comprehensive analysis of computational efficiency metrics across different feature dimensions, showing rendering speed (FPS), storage size, GPU memory usage, and runtime performance as Gaussian size and frame numbers vary.

Method	Ramen	Figurines	Teatime	Waldo Kitchen	Overall
Ours	51.0	58.6	70.2	65.3	61.3
(w/o) wtd. agg	45.3	56.7	70.0	64.8	59.9
(w/o) filter	47.9	57.1	67.7	62.3	58.8
(w/o) wtd. agg + filter	44.1	54.9	69.9	59.6	57.1

Table 4: Ablation study on the impact of filtering and weighted aggregation on feature learning. Evaluate on LERF dataset and use average IoU as the metric for comparison.

represents a two orders of magnitude improvement in processing speed while delivering superior performance across test datasets without feature compression.

We assess our method's scalability across scene complexity and feature dimensions, focusing on runtime, storage, rendering speed, and GPU memory requirements. As shown in Figure 5, our approach maintains high training efficiency even with scenes containing up to 4 million Gaussians (d), while keeping memory requirements below 14GB for 512D CLIP features (a) - within the capabilities of most modern GPUs. Our method's flexibility supports arbitrary feature dimensions, presenting clear trade-offs: higher dimensions reduce rendering speed (c) and increase storage requirements (b). These findings enable users to balance quality and performance based on available resources and application needs.

4.7 Ablation Study

Our ablation study in Table 4 validates two key components of our method. First, accurately modeling the forward rendering process via weighted feature aggregation (w/o wtd. agg) significantly improves multi-view fusion by accurately modeling how features should blend based on Gaussian visibility. Second, filtering inactive Gaussians (w/o filter) enhances performance by eliminating occluded Gaussians that create semantic inconsistencies—these problematic Gaussians (with influence $w_i^s < 10^{-4}$) may minimally affect neighboring pixels during rendering without receiving proper feature updates, causing visual noise. Removing both components reduces performance by 4.2%, confirming that each element meaningfully contributes to our method's effectiveness through their complementary improvements to feature consistency and computational efficiency.

5 Discussion and Conclusion

In this work, we presented a theoretically grounded and optimization-free approach for efficient Language Gaussian Splatting. The proposed method shows state-of-the-art results with a single iteration over all views and avoids rendering and loss computation in the feature space. Beyond the immediate performance gains, our approach opens up new possibilities for how we approach 3D scene understanding and manipulation. The ability to reason directly in language feature space, without compression, enables practical applications like object insertion and scene registration that were previously computationally prohibitive. Overall, we demonstrated that: **1)** High-dimensional language features can directly be used in Gaussian splatting by utilizing a training-free optimization approach. **2)** Accurate modeling of the rendering process enables the uplift of highly relevant features from 2D to 3D.

6 Limitation

Our method relies on several simplifying assumptions that may limit its applicability in certain scenarios. First, we assume that semantic features exhibit spatial coherence and that alpha-blending weights provide a reasonable proxy for semantic visibility, which may not hold in highly cluttered scenes with complex occlusions or significant semantic discontinuities. Second, as a training-free approach, our method's performance is fundamentally bounded by the quality of the underlying 2D feature extractor (e.g., CLIP) and cannot adapt or refine semantics for scene-specific content. Future work could explore incorporating a lightweight learning component to fine-tune features for specific 3D scenes while maintaining computational efficiency

References

[1] Yash Bhalgat, Iro Laina, João F Henriques, Andrew Zisserman, and Andrea Vedaldi. N2f2: Hierarchical scene understanding with nested neural feature fields. *arXiv preprint arXiv:2403.10997*, 2024.

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[3] Runnan Chen, Youquan Liu, Lingdong Kong, Xinge Zhu, Yuexin Ma, Yikang Li, Yuenan Hou, Yu Qiao, and Wenping Wang. Clip2scene: Towards label-efficient 3d scene understanding by clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7020–7030, 2023.

[4] Runyu Ding, Jihan Yang, Chuhui Xue, Wenqing Zhang, Song Bai, and Xiaojuan Qi. Pla: Language-driven open-vocabulary 3d scene understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7010–7019, 2023.

[5] 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.

[6] Rahul Goel, Dhawal Sirikonda, Saurabh Saini, and PJ Narayanan. Interactive segmentation of radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4201–4211, 2023.

[7] Qiao Gu, Alihusein Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, Chuang Gan, Celso Miguel de Melo, Joshua B. Tenenbaum, Antonio Torralba, Florian Shkurti, and Liam Paull. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning, 2023. URL https://arxiv.org/abs/2309.16650.

[8] Jun Guo, Xiaojian Ma, Yue Fan, Huaping Liu, and Qing Li. Semantic gaussians: Open-vocabulary scene understanding with 3d gaussian splatting, 2024.

[9] Saurabh Gupta, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Indoor scene understanding with rgb-d images: Bottom-up segmentation, object detection and semantic segmentation. *International Journal of Computer Vision*, 112:133–149, 2015.

[10] Huy Ha and Shuran Song. Semantic abstraction: Open-world 3d scene understanding from 2d vision-language models. *arXiv preprint arXiv:2207.11514*, 2022.

[11] Ayaan Haque, Matthew Tancik, Alexei A. Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19740–19750, October 2023.

[12] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

[13] 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.

[14] 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.

[15] 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.

[16] Feng Liang, Bichen Wu, Xiaoliang Dai, Kunpeng Li, Yinan Zhao, Hang Zhang, Peizhao Zhang, Peter Vajda, and Diana Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7061–7070, 2023.

[17] 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.

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

[19] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024.

[20] Qucheng Peng, Benjamin Planche, Zhongpai Gao, Meng Zheng, Anwesa Choudhuri, Terrence Chen, Chen Chen, and Ziyan Wu. 3d vision-language gaussian splatting. *arXiv preprint arXiv:2410.07577*, 2024.

[21] Qucheng Peng, Benjamin Planche, Zhongpai Gao, Meng Zheng, Anwesa Choudhuri, Terrence Chen, Chen Chen, and Ziyan Wu. 3d vision-language gaussian splatting, 2024. URL https://arxiv.org/abs/2410.07577.

[22] Songyou Peng, Kyle Genova, Chiyu "Max" Jiang, Andrea Tagliasacchi, Marc Pollefeys, and Thomas Funkhouser. Openscene: 3d scene understanding with open vocabularies, 2023. URL https://arxiv.org/abs/2211.15654.

[23] 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.

[24] Yansong Qu, Shaohui Dai, Xinyang Li, Jianghang Lin, Liujuan Cao, Shengchuan Zhang, and Rongrong Ji. Goi: Find 3d gaussians of interest with an optimizable open-vocabulary semantic-space hyperplane, 2024. URL https://arxiv.org/abs/2405.17596.

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[26] Antoni Rosinol, Arjun Gupta, Marcus Abate, Jingnan Shi, and Luca Carlone. 3d dynamic scene graphs: Actionable spatial perception with places, objects, and humans, 2020. URL https://arxiv.org/abs/2002.06289.

[27] Qiuhong Shen, Xingyi Yang, and Xinchao Wang. Flashsplat: 2d to 3d gaussian splatting segmentation solved optimally. *European Conference of Computer Vision*, 2024.

[28] Yunhang Shen, Chaoyou Fu, Peixian Chen, Mengdan Zhang, Ke Li, Xing Sun, Yunsheng Wu, Shaohui Lin, and Rongrong Ji. Aligning and prompting everything all at once for universal visual perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13193–13203, 2024.

[29] 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.

[30] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Superfast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5459–5469, 2022.

[31] Ayça Takmaz, Elisabetta Fedele, Robert W Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. Openmask3d: Open-vocabulary 3d instance segmentation. *arXiv preprint arXiv:2306.13631*, 2023.

[32] Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models, 2024. URL https://arxiv.org/abs/2402.12289.

[33] Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. Neural feature fusion fields: 3D distillation of self-supervised 2D image representations. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2022.

[34] Yu Wang, Xiaobao Wei, Ming Lu, and Guoliang Kang. Plgs: Robust panoptic lifting with 3d gaussian splatting, 2024. URL https://arxiv.org/abs/2410.17505.

[35] 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.

[36] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *European Conference on Computer Vision*, pages 162–179. Springer, 2025.

[37] Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey Hu, Matthew Tancik, and Angjoo Kanazawa. gsplat: An open-source library for Gaussian splatting. *arXiv preprint arXiv:2409.06765*, 2024. URL https://arxiv.org/abs/2409.06765.

[38] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. *arXiv preprint arXiv:2112.05131*, 2(3):6, 2021.

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

[40] Junbo Zhang, Runpei Dong, and Kaisheng Ma. Clip-fo3d: Learning free open-world 3d scene representations from 2d dense clip. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2048–2059, 2023.

[41] Hui Zhou, Xinge Zhu, Xiao Song, Yuexin Ma, Zhe Wang, Hongsheng Li, and Dahua Lin. Cylinder3d: An effective 3d framework for driving-scene lidar semantic segmentation. *arXiv preprint arXiv:2008.01550*, 2020.

[42] 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.

[43] 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.