Language Embedded 3D Gaussians for Open-Vocabulary Scene Understanding

Jin-Chuan Shi[1] Miao Wang[1,2*] Hao-Bin Duan[1] Shao-Hua Guan[1]
[1]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University
[2]Zhongguanchun Laboratory

https://buaavrcg.github.io/LEGaussians

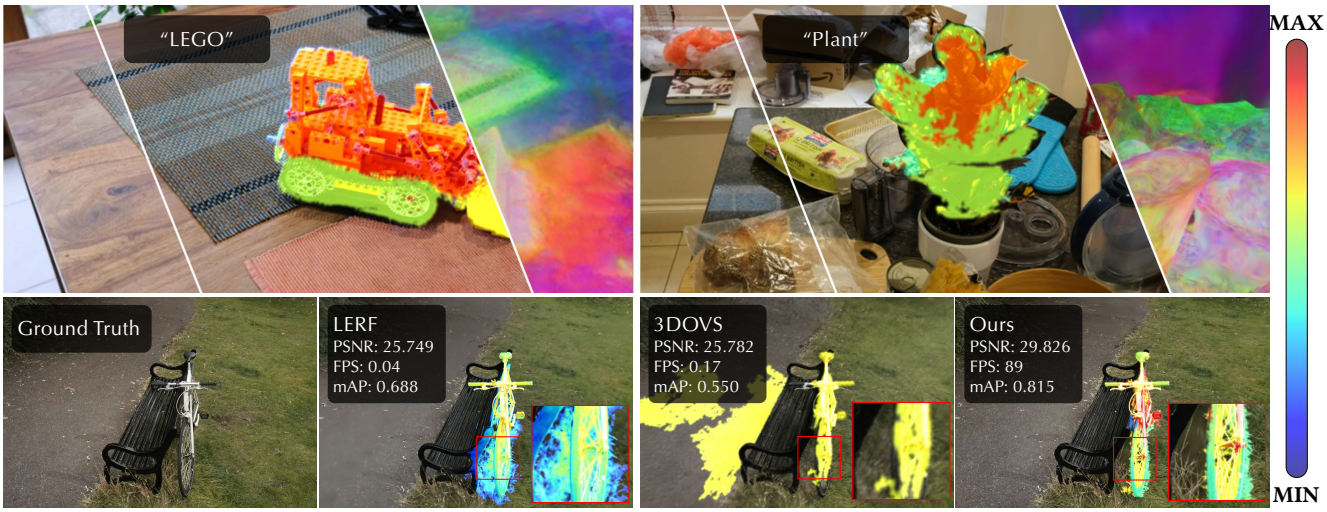

Figure 1. We present Language Embedded 3D Gaussians, a novel scene representation for open-vocabulary querying. The top row visualizes the original image, novel view synthesis result with query relevancy and PCA of learned semantic features. The bottom row compares our method with other language-embedded representations. The right-side bar maps relevancy values to heatmap colors. Our method achieves better fidelity and query accuracy while rendering at higher frame rates.

Abstract

Open-vocabulary querying in 3D space is challenging but essential for scene understanding tasks such as object localization and segmentation. Language-embedded scene representations have made progress by incorporating language features into 3D spaces. However, their efficacy heavily depends on neural networks that are resource-intensive in training and rendering. Although recent 3D Gaussians offer efficient and high-quality novel view synthesis, directly embedding language features in them leads to prohibitive memory usage and decreased performance. In this work, we introduce Language Embedded 3D Gaussians, a novel scene representation for open-vocabulary query tasks. Instead of embedding high-dimensional raw semantic features on 3D Gaussians, we propose a dedicated quantization scheme that drastically alleviates the memory requirement, and a novel embedding procedure that achieves smoother yet high accuracy query, countering the multi-view feature inconsistencies and the high-frequency inductive bias in point-based representations. Our comprehensive experiments show that our representation achieves the best visual quality and language querying accuracy across current language-embedded representations, while maintaining real-time rendering frame rates on a single desktop GPU.

* Corresponding author.

1. Introduction

Neural Radiance Field (NeRFs) [2, 3, 28, 50] and 3D Gaussian Splatting [17] has advanced the development of efficient and high-quality 3D scene novel view synthesis from multi-view images. Nevertheless, these representations solely encapsulate geometric and appearance details without any semantic information. To bridge this gap, language-embedded neural representations [18, 19] try to integrate se-

mantic information from multi-view images into 3D scenes for open-vocabulary querying tasks, which allows intuitive interaction with large language models (LLMs) [47, 48] and human users, and powers broad applications including scene editing, VR/AR, autonomous driving, and robotic navigation. Compared to traditional semantic labeling methods, language features from visual-language models like CLIP [32] and DINO [5] offer more comprehensive semantic understanding capability, as they encompass objects with long-tail distribution, enhancing their suitability for real-world applications. However, accurately incorporating language embedding into current 3D scene representations, while maintaining their efficiency and visual quality, presents a significant challenge.

Recent techniques [18, 19, 23] extract dense language features from multi-view 2D images and incorporate additional output branches in scene representation to predict semantic features. However, the quality of semantic features heavily relies on scene representation, and trivially expanding the output channels poses significant challenges in recovering high-precision and robust semantics of the scenes. Furthermore, while 3D Gaussian [17] are efficient and fast for high-quality 3D scene representation, embedding raw semantic information into a massive number of points can cause prohibitive memory requirements and significantly lower the efficiency of both optimization and rendering.

In this paper, we introduce Language Embedded 3D Gaussians, a semantic scene representation framework that provides both high precision and efficiency for open-vocabulary query tasks. To avoid the substantial memory requirement that would result from directly increasing the feature channels for raw semantic features in the 3D Gaussians, we employ a novel feature quantization approach that significantly decreases the computational and memory cost. By leveraging the redundancy nature of local semantic features, we construct more streamlined language features that are compactly stored on the 3D Gaussians. Additionally, we address the issue of semantic ambiguity caused by visual inconsistency from multi-view images by implementing a mechanism that lowers the spatial frequency of semantic features, guided by learned uncertainty values. This technique enables language features in the 3D Gaussians to be much smoother yet still precise.

Our extensive experiments demonstrate that our method achieves state-of-the-art quality in both novel view synthesis and open-vocabulary querying tasks, while allowing real-time rendering on consumer-level devices.

In summary, our contributions include:

- We introduce a novel quantization scheme that efficiently compresses and integrates semantic features into dense 3D Gaussians, ensuring efficient optimization and rendering on consumer devices while maintaining accurate semantic embedding.
- We propose a mechanism that leverages spatial position and semantic uncertainty of 3D Gaussians to address the semantic ambiguity arising from visual inconsistency across views.
- Our method outperforms other language-embedded 3D representations, delivering state-of-the-art results in visual quality, language query precision, and rendering speed at the same time.

2. Related Work

Neural Rendering. NeRF [28] has demonstrated superior novel view synthesis quality over traditional methods [6, 9, 14, 15, 35], and various efforts have been made to enhance its performance. Although many methods focus on improving NeRF's rendering quality [2, 3, 27, 33, 38, 50], they still suffer from slow training and rendering speed. On the other hand, explicit and hybrid scene representations [4, 7, 12, 24, 29, 36, 44]. These methods typically utilize hash grids [29] and point clouds [44] to reduce computational cost of large neural networks. Recent 3D Gaussian [17] sets a new standard on both rendering quality and training speed, by using fast rasterization of 3D Gaussians to replace differential volume rendering. Nevertheless, directly embedding language embeddings on the dense 3D Gaussians is non-trivial, as the massive amount of 3D points requires prohibitive memory usage and causes a drastic performance drop in both training and rendering.

Language Embedded Scene Representation. Incorporating specific semantics into NeRF's implicit MLP-based representation is difficult due to challenges in accurately identifying 3D regions. Various approaches [10, 13, 18, 19, 34, 39, 41, 54] have tried to integrate semantic data into NeRF, often by adding new network branches. For instance, DFF [19] embeds a branch for language predictions and uses a pre-trained encoder for supervision, while LERF [18] assimilates CLIP [32] features from multi-scale image crops into hash grid-represented 3D scenes. 3D-OVS [22] takes a different route, creating dual decomposed 3d tensors for geometry and semantics, but it requires predefined segmented classes and doesn't support arbitrary queries. Other methods [20, 53] decompose scenes into smaller MLPs representing local semantics. However, the quality of semantic embeddings and the efficiency of training and rendering of these methods, are all constrained by the limitations of their scene representation models.

Open Vocabulary 3D Scene Understanding. Recent progress in open vocabulary scene understanding has been marked by the integration of 2D Vision-Language Models with 3D point cloud processing [16, 45, 46, 52, 55]. These approaches focus on aligning features and projecting

3D data into 2D to enhance zero-shot learning capabilities. Additionally, advancements in 3D object detection and segmentation [8, 26, 31, 37, 49] have shown the effectiveness of merging point cloud data with visual features extracted from image for scene analysis [43]. However, these methods mainly address the comprehension and analysis of existing scene representations, like point clouds, rather than optimizing scene representation from multi-view images, which is the focus of our work.

3. Method

In this section, we introduce our training process of Language Embedded 3D Gaussians, including (1) a recap of 3D Gaussian Splatting [17] (Sec. 3.1), (2) extracting dense language features from multi-view images (Sec. 3.2), (3) a quantization scheme for high-dimensional language features that creates a compact feature space for the embedding process (Sec. 3.3), (4) an embedding mechanism that alleviates semantic ambiguity caused by visual difference across views, by lowering the spatial frequency of semantic features with the learned uncertainty values (Sec. 3.4).

3.1. Recap: 3D Gaussian Splatting

3D Gaussian Splatting [17] renders complex scenes by merging a multitude of colored 3D Gaussians, which are subsequently projected onto camera views through a rasterization process. By employing differentiable rendering and gradient descent, the attributes of these 3D Gaussians, including position p, covariance Σ, color c, and opacity α, are optimized to represent the 3D scene based on a collection of input images. The result image I is rendered from a specific camera pose p_{cam}, using the differentiable rasterization R, represented as:

$$I = R(p, \Sigma, c, \alpha; p_{\text{cam}}). \tag{1}$$

3.2. Dense Language Feature Extraction

We first extract pixel-level dense language features from visual-language models. While CLIP [32] encodes images into global semantic features, its direct application is not feasible for our purposes as we require pixel-level targets to learn 3D scene representations from multi-view images. Prior studies [18, 22] overcome this limitation by computing multi-scale dense CLIP features for each image. To obtain dense language embeddings from multi-view images, we employ a slightly different hierarchical random cropping technique to extract CLIP features, similar to 3DOVS [22]. We aggregate features from all layers and normalize them to produce the final language embeddings.

Nevertheless, features extracted from CLIP only provide a rough boundary of different semantic regions, resulting in ambiguities and inaccuracies in the language embedding of 3D scenes. Conversely, DINO [5] exhibits autonomous object decomposition without requiring labeled data [1], as several studies [18, 19, 22, 42] have shown that DINO effectively enhances language feature grouping without reliance on labels or prior knowledge. Therefore, we also extract DINO features as complementary to enhance the details of extracted language features.

We then concatenate the dense CLIP and DINO features extracted from multi-view images to form hybrid language feature maps. We denote the features at the position (x, y) on the CLIP and DINO feature maps of image I as $\mathbf{F}^{\text{CLIP}}_{I,x,y} \in \mathbb{R}^{d_{\text{CLIP}}}$ and $\mathbf{F}^{\text{DINO}}_{I,x,y} \in \mathbb{R}^{d_{\text{DINO}}}$, respectively, where d_{CLIP} and d_{DINO} represent the dimension of each feature. The hybrid language feature $\mathbf{F}_{I,x,y} \in \mathbb{R}^d$ is given by:

$$\mathbf{F}_{I,x,y} = \mathbf{F}^{\text{CLIP}}_{I,x,y} \oplus \mathbf{F}^{\text{DINO}}_{I,x,y}, \tag{2}$$

where d is the total channels in the hybrid features.

3.3. Quantization of Language Features

Direct integration of our extracted language features as in prior studies [18, 19, 22] would be infeasible for 3D Gaussians, due to the substantial memory demands of recording high-dimensional language embeddings on them, which significantly hinders rendering efficiency and limits the maximum number of 3D Gaussians that can be trained at the same time. To alleviate such storage and computation costs of raw language embeddings, our insight is to leverage the inherent redundancy in language features, as those within a single object share very similar semantics. Additionally, the semantics of a single scene merely cover only a small fraction of the original CLIP feature space, leading to unnecessary data stored in the 3D representation for all the queries we would need. To this end, we propose a dedicated quantization scheme to effectively compress the semantic features extracted from multiple viewpoints, resulting in a more efficient and compact representation of scene-aware semantic features.

Specifically, our goal is to transform the hybrid language features $\mathbf{F} \in \mathbb{R}^d$ into a quantized version $\hat{\mathbf{F}} \in \mathbb{R}^d$, which approximates $\mathbf{F}$. To achieve this, we develop a discrete language feature space $\mathcal{S} = \{\mathbf{f}_i \in \mathbb{R}^d | i = 1, 2, \cdots, N\}$, and use an integer index $m \in \{1, 2, \cdots, N\}$ for retrieving the nearest language feature in $\mathcal{S}$. The quantized language feature is then given by:

$$\hat{\mathbf{F}} = \sum_i^N \mathbf{f}_i \cdot \text{onehot}(m)_i. \tag{3}$$

Here, $\mathbf{f}_i$ represents the i-th basis of the language features, and $\text{onehot}(m) \in \mathbb{R}^N$ is the one-hot vector for index m. This quantization scheme effectively removes excessive information in the original language embeddings by compressing the original continuous language feature space into discrete bases, with the compression rate adjustable through the size of space N.

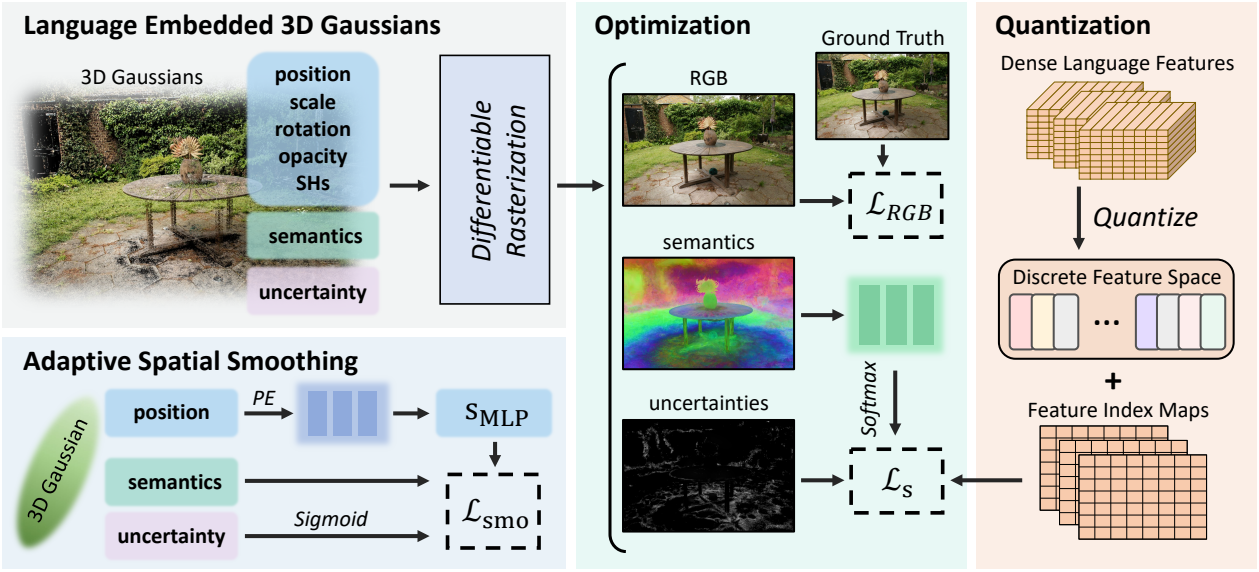

Figure 2. The training process for Language-embedded 3D Gaussians starts with initializing scenes following 3D Gaussian Splatting [17] and randomly initializing semantic features and setting uncertainty to zero. Dense language features from multi-view CLIP [32] and DINO [5] are quantized to create a discrete feature space and semantic indices. These attributes of the 3D Gaussians are then rendered into 2D maps using a differentiable rasterizer. The optimization is achieved through semantic and adaptive spatial smoothing loss.

Quantization. To quantitize high-dimentinal language feature $\mathbf{F}$, we execute a max-similarity search within space $\mathcal{S}$, utilizing cosine similarity $\cos\langle\cdot\rangle$ as the distance metric:

$$\mathcal{D}(\mathbf{F}, \mathbf{f}_i) = cos\langle \mathbf{F}^{\text{CLIP}} \cdot \mathbf{f}_i^{\text{CLIP}} \rangle + \lambda_{\text{DINO}} cos\langle \mathbf{F}^{\text{DINO}} \cdot \mathbf{f}_i^{\text{DINO}} \rangle, \tag{4}$$

where λ_{DINO} serves as a hyper-parameter that modulates the significance of DINO within the semantic feature.

Following VQ-VAE [40], the selected language feature index from the set $\mathcal{S}$ is determined as $m = \text{argmax}_i(\mathcal{D}(\mathbf{F}, \mathbf{f}_i))$, and the quantization of $\mathbf{F}$ is computed using Eq. (3). The result for each image after this quantization procedure is a semantic indices map, denoted as $\mathcal{M} \in \mathbb{R}^{H \times W \times 1}$.

Optimization. During the quantization of all language features extracted from multi-view images, the optimization of the discrete feature space $\mathcal{S}$ is simultaneously accomplished by minimizing the cosine similarity loss between the language features $\mathbf{F}_i$ and the quantization $\hat{\mathbf{F}}_\mathbf{i}$:

$$\begin{aligned} \mathcal{L}_{cos}(\mathbf{F}_i) =& (1 - cos\langle \mathbf{F}_i^{\text{CLIP}} \cdot \hat{\mathbf{F}}_i^{\text{CLIP}} \rangle) \\ &+ \lambda_{\text{DINO}} (1 - cos\langle \mathbf{F}_i^{\text{DINO}} \cdot \hat{\mathbf{F}_i^{\text{DINO}}} \rangle). \end{aligned} \tag{5}$$

Furthermore, to prevent quantization collapse and ensure maximal utilization of each feature in the feature space, we have devised a load balancing loss inspired by the design of the Switch Transformer [11]. The load balancing loss is calculated by performing an element-wise multiplication of the utilization ratio $\mathbf{r} \in \mathbb{R}^N$ and the mean selection probability $\mathbf{p} \in \mathbb{R}^N$ of each feature, followed by their summation:

$$\mathcal{L}_{lb} = \sum^{N} (\mathbf{r} \circ \mathbf{p}), \tag{6}$$

where $\circ$ represents the element-wise product. We provide a detailed description of the load balancing loss in the supplementary material.

In summary, when quantizing the multi-view semantic features, we optimize the discrete feature space $\mathcal{S}$ as well as the semantic indices maps $\mathcal{M}$ using the following loss:

$$\mathcal{L}_q = \lambda_{cos} \mathcal{L}_{cos} + \lambda_{lb} \mathcal{L}_{lb}. \tag{7}$$

3.4. Language Embedded 3D Gaussians

Utilizing the discrete feature space $\mathcal{S}$ and index maps $\mathcal{M}$ in the previous section, we embed the compressed language features into 3D Gaussians for open-vocabulary scene understanding. Concretely, we expand the number of channels of each 3D Gaussian to include a compact feature vector representing the discrete index of language feature bases. To address semantic ambiguity arising from visual disparities across various viewpoints, we introduce a novel mechanism to reduce the spatial frequency of language embeddings through an adaptive learning loss based on the learned uncertainty values on each point.

3.4.1 Compact Semantic Features on 3D Gaussians

The inherent attributes of 3D Gaussians, such as color, are rendered onto the screen through the processes of rasterization and alpha blending (see Sec. 3.1). However, embedding discrete semantic indices m onto 3D Guassians would

lead to erroneous results when optimizing through differential rendering, as these indices are not in a continuous space. Instead of directly embedding indices, we learn another continuous and compact semantic feature vectors, denoted as $\mathbf{s}_G \in \mathbb{R}^{d_s}$, where d_s is a hyper-parameter for controlling the storage capability for semantics. We then render these compact semantic feature vectors into a 2D feature map with rasterization and alpha blending, and decode the 2D feature map into the discrete semantic indices m using a tiny MLP decoder:

$$\hat{\mathcal{M}} = \text{softmax}(D_{\text{MLP}}(R_s(\mathcal{G}; p_{\text{cam}}))), \tag{8}$$

where $R_s(\mathcal{G}; p_{\text{cam}}) \in \mathbb{R}^{H \times W \times d_s}$ represents the semantic feature rendering from a set of 3D Gaussians $\mathcal{G}$ viewed from camera pose p_{cam}. D_{MLP} represents the tiny MLP decoder. During training process, a softmax operation is applied to the decoder's output, yielding the semantic feature index distribution $\hat{\mathcal{M}} \in \mathbb{R}^{H \times W \times N}$, where H and W denote the height and width of the image, respectively. To optimize the semantic features of the 3D Gaussians and the MLP decoder, we apply the cross-entropy loss:

$$\mathcal{L}_{\text{CE}} = \text{CE}(\hat{\mathcal{M}}, \mathcal{M}), \tag{9}$$

where $\mathcal{M} \in \mathbb{R}^{H \times W \times 1}$ denotes the discrete semantic indices map of the image extracted during the language feature quantization process (Sec. 3.3).

3.4.2 Semantic Feature Smoothing

Due to viewing angles, illuminations, existence of specular and semi-transparent materials, language features of the same spatial location in multi-view images may exhibit high variance, posing a challenge for precise semantic learning in 3D scenes. The high variance in language features at the same spatial location in multi-view images, due to differences in viewing angles, illumination, and the presence of specular and semi-transparent materials, presents challenges in accurately learning semantics in 3D scenes. Moreover, partially occluded objects may have their semantic features inaccurately extracted due to the lack of detection of their entirety. When these biased semantic features undergo quantization, the error may be amplified, resulting in a single 3D position being linked to several distinct language feature bases and, therefore, different discrete feature indices. To this end, we introduce a smoothing strategy that limits the spatial frequency of semantic features on 3D Gaussians. This is achieved by incorporating an adaptive loss based on a learnable uncertainty value for each 3D Gaussian.

Learning of uncertainty. To represent the variance associated with the semantic feature for each Gaussian, we record an optimizable semantic uncertainty on each point, denoted as $u \in [0, 1]$. A higher u suggests that the semantic feature may exhibit instability and undergo frequent changes during the optimization process. The uncertainty values are jointly optimized when training the compact semantic features $\mathbf{s}_G$:

$$\mathcal{L}_{\text{CE}} = \frac{\sum \text{CE}(\hat{\mathcal{M}}, \mathcal{M}) \circ (1 - R_u(\mathcal{G}; p_{\text{cam}}))}{H \times W}, \tag{10}$$

where $R_u(\mathcal{G}; p_{\text{cam}}) \in \mathbb{R}^{H \times W \times 1}$ is the rendered 2D uncertainty map from camera pose p_{cam}. At the same time, we regularize these uncertainty values to avoid converging to a trivial solution where all 3D Gaussians have the maximum uncertainties:

$$\mathcal{L}_u = \frac{\sum R_u(\mathcal{G}; p_{\text{cam}})}{H \times W}. \tag{11}$$

In summary, the total semantic loss for optimizing the compact semantic features of the 3D Gaussians and the MLP decoder is defined as:

$$\mathcal{L}_s = \lambda_{\text{CE}} \mathcal{L}_{\text{CE}} + \lambda_u \mathcal{L}_u, \tag{12}$$

where λ_u is a hyper-parameter for controlling the weight of regularization. We initialize the uncertainty values to zero at the start of training, and inconsistent semantic feature indices will lead to increased u during optimization.

Adaptive spatial smoothing loss. Building on the earlier observation that spatially adjacent positions typically display similar semantic features, we utilize this prior knowledge to deliberately decrease the spatial frequency of the embedded compact semantic features in 3D Gaussians, particularly for those with high uncertainty values. We harness the inductive biases of coordinate-based MLPs, known for learning low-frequency representations of target signals, to regularize the semantic features in 3D Gaussians. We calculate the smoothed semantic features $\mathbf{s}_{\text{MLP}} \in \mathbb{R}^{d_s}$ by inputting the position of each 3D Gaussian into a small MLP:

$$\mathbf{s}_{\text{MLP}} = \text{MLP}(\text{PE}(\mathbf{p})), \tag{13}$$

where $\mathbf{p} \in \mathbb{R}^3$ represents the position of Gaussian, and PE represents the positional encoding used in NeRF [28]. We set a low frequency in the positional encoding to encourage spatial smoothness. We then apply the following loss for imposing the spatial smoothness regularization, where the degree of smoothness is adaptively controlled based on the learned uncertainty values:

$$\mathcal{L}_{\text{smo}} = \|\mathbf{s}_{\text{MLP}} - \mathbf{s}^*_G\|_2 + \max(\mathbf{u}^*_G, w_s) \|\mathbf{s}^*_{\text{MLP}} - \mathbf{s}_G\|_2, \tag{14}$$

where the $*$ denotes the stop gradient operator, and w_s is the minimal weight for the semantic smoothing term.

Method	PSNR↑	SSIM↑	LPIPS↓	mPA↑	mP↑	mIoU↑	mAP↑	FPS↑	Memory↓	Storage↓	Training Time↓
DFF [19]	25.378	0.712	0.312	0.817	0.124	0.091	0.199	0.202	42GB+14GB	41GB	184min
LERF [18]	25.749	0.811	0.317	0.890	0.475	0.403	0.688	0.04	25GB+6GB	320MB	**54min**
3DOVS [22]	25.782	0.733	0.295	0.905	0.529	0.458	0.550	0.17	57GB+15GB	205GB	158min
Ours	**29.826**	**0.901**	**0.112**	**0.947**	**0.753**	**0.578**	**0.815**	**89**	**11GB+12GB**	**15MB**	68min

Table 1. Quantitative comparison of our method with DFF [19], LeRF [18], 3DOVS [22].

		mPA↑	mP↑	mIoU↑	mAP↑
Quantization	w/o DINO	0.927	0.604	0.481	0.676
	w/o $\mathcal{L}_{lb}$	0.939	0.666	0.544	0.738
Embedding	w/o u_G	0.944	0.717	0.576	0.773
	w/o MLP	0.945	0.715	**0.580**	0.774
	w/o u_G & MLP	0.944	0.717	**0.580**	0.774
Ours	-	**0.947**	**0.753**	0.578	**0.815**

Table 2. Quantitative results of ablation experiments.

To summarize, the combined loss for optimization of semantic features, semantic uncertainties, and MLPs is given by:

$$\mathcal{L} = \lambda_s \mathcal{L}_s + \lambda_{\text{smo}} \mathcal{L}_{\text{smo}}. \tag{15}$$

4. Implementation Details

We implement our method using PyTorch [30], and incorporate the CUDA kernel from 3D Gaussian Splatting [17] to speed up the rasterization rendering process. Our method optimizes the scene's geometry and appearance with the same RGB loss following 3D Gaussian Splatting and enables adaptive density control of 3D Gaussians during the reconstruction process. We modify the CUDA kernel to enable the rendering of semantic features on the 3D Gaussians, and ensure that the optimizing of these semantic parameters does not affect original reconstruction quality. We set $d_s = 8$ and $w_S = 0.1$ in our modified 3D Gaussians, and set $\lambda_D = \lambda_{lb} = 0.5$, and all other λ values are set to 1. After the phase of extracting dense semantic features, which takes about 30 minutes, our model can be trained on one RTX3090 GPU for about 1 hour. The training involves 30,000 iterations, utilizing the Adam optimizer [25], with a learning rate set to 0.001 and betas equal to $(0.9, 0.999)$. We leave the inference details of open-vocabulary language querying and semantic relevancy calculation in the supplementary material.

5. Experiments

5.1. Basic Setups

Dataset. For a simultaneous evaluation of visual and semantic embedding quality, we select six scenes (excluding Stump) from the Mip-NeRF360 dataset [3] and manually annotate segmentation maps for each scene in the evaluation set. Each scene encompasses 180 to 320 images captured from various angles, and the evaluation set is chosen randomly with many novel-view images. Segmentation masks are annotated for primary objects in each scene.

Baseline Methods and Metrics. We conduct a comparative evaluation of our method with DFF [19], LeRF [18], and 3DOVS [22], focusing on visual quality, language-embedded accuracy, rendering speed and model efficiency. To measure the visual quality in novel views, we report the PSNR, SSIM, and LPIPS [51] metrics. For the accuracy of language embedding, we measure the mean intersection over union (mIoU), mean pixel accuracy (mPA), mean precision (mP), and mean average precision (mAP) based on our annotations. The rendering speed (FPS) is measured by rendering the images with language features at a consistent resolution. Additionally, model efficiency is evaluated based on CPU and GPU memory usage during training, as well as data storage requirements and training duration.

5.2. Comparisons

We compare our method both qualitatively and quantitatively with DFF [19], LeRF [18], and 3DOVS [22] on our annotated Mip-NeRF360 dataset using a single RTX3090 GPU, following their default parameters but at the same resolution as our method.

Qualitative Results. Fig. 9 displays a qualitative comparison of the novel view synthesis and semantic embedding results, demonstrating our method's efficacy in querying challenging objects in both indoor and outdoor scenes. Our approach notably delivers the highest visual rendering quality and query accuracy across all the tested scenes. Specifically, DFF [19] fails to identify "asphalt ground" in scene "bicycle" and "flower" in scene "garden". This may be caused by its use of LSeg [21], which is unstable to compute correct features in complex scenes. Moreover, due to predetermined query categories during training, 3DOVS [22] shows poor performance in scenes with complex objects. While LERF [18] can locate queried objects, its grid-based scene representation limits its ability to define clear boundaries. In contrast, our point-based approach supports high-frequency embedded semantic features, allowing for enhanced spatial semantic accuracy simply by incorporating more 3D Gaussians into scenes. This is facilitated by our quantization method, which substantially lowers memory

Figure 3. Comparison of novel view synthesis and query relevance visualization. Left to right: Ground truth novel view synthesis, novel view images with relevance visualization from our method, DFF [19], LeRF [18], and 3DOVS [22]. Top to bottom: Query words "asphalt ground", "bicycle", "jar of coconut oil", "flower", "LEGO Technic 856 Bulldozer", and "brown shoes".

costs. Additionally, our adaptive semantic smoothing technique effectively manages the high variance and ambiguity observed from different viewpoints, selectively applying spatial consistency as required.

Quantitative Results. Tab. 1 presents a comparison across various metrics, including novel view synthesis quality, open-vocabulary query accuracy, and computational efficiency. We report both host memory and video memory usage, as well as the disk space used for storing the learned language features. Our approach outperforms others in rendering quality and semantic query accuracy, while also offering lower computational demands and a significant speed increase, nearly 100 times faster in inference. It's noteworthy that methods like 3DOVS and DFF require extensive memory and storage due to their use of raw language features during training. In contrast, our use of the quantization scheme facilitates the incorporation of detailed semantics into complex 3D scenes with numerous 3D Gaussians, and concurrently achieves the most efficient storage utilization among the baseline methods.

5.3. Open-vocabulary Query

Fig. 5 illustrates the results of different open-vocabulary queries. We use a diverse range of vocabulary categories to identify objects in scenes, such as visual attribute terms like "green", and subjective adjectives like "cute". Furthermore, the figure shows our method's effectiveness in identifying both large-scale objects like "lego" and specific components of an object, such as a "engine". We show more

Figure 4. Visual quality comparison of novel view synthesis results. Our method is able to recover more detailed geometry and appearance compared to other methods.

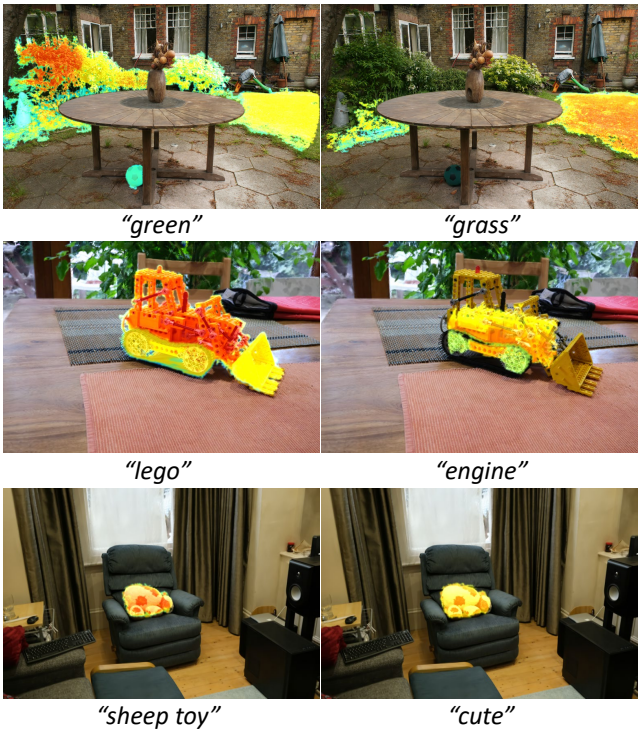

Figure 5. Images of various open-vocabulary queries.

examples in the supplementary material.

5.4. Ablation Study

We demonstrate the results of ablation studies in Tab. 2 and Fig. 6. The results show that embedding uncertainty without spatial smoothing of semantic features leads to suboptimal optimization. Conversely, using MLP solely for spatial smoothing, without accounting for semantic variance across multiple views, also impedes the optimization of precise language features. However, our full model, combining uncertainty with MLP smoothing in an adaptive manner effectively diminishes ambiguity and enhances the mean average precision (mAP) metric. Furthermore, integrating DINO features significantly improves the definition of object query boundaries. The load balancing loss, introduced during the quantization phase, results in a more utilized discrete feature space, facilitating the distinguish of objects with similar semantics, thereby boosting overall accuracy.

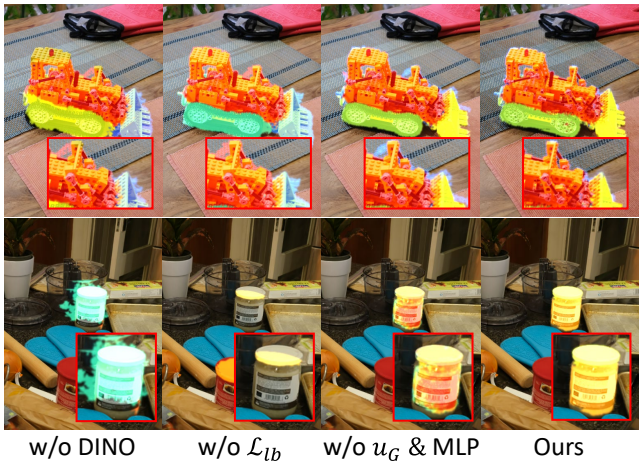

Figure 6. Comparison of ablation experiments.

6. Conclusion

We present Language Embedded 3D Gaussians, a novel scene representation designed for open-vocabulary query tasks. Our method successfully embeds quantized compact semantics features onto massive 3D Gaussians, while only maintaining minimal memory and storage requirements. To address semantic inconsistencies across different viewpoints, we propose a feature smoothing procedure that adaptively lowers the spatial frequency of embedded semantic features, guided by the uncertainty values learned on the 3D Gaussians. The result is a highly effective scene representation that not only enables high-quality novel view synthesis but also provides high accuracy in open-vocabulary querying, all achieved with modest computational resources.

Limitations and Future works. Although DINO features improve object boundary detection, they fall short in pinpointing fine-grained object geometries at high resolutions when using CLIP-derived semantics. Additionally, detecting highly reflective or translucent objects, such as televisions and mirrors, remains challenging. These limitations might be overcome with more

advanced visual-language models and native per-pixel semantic features. Nevertheless, our approach can be further adapted for broader open-vocabulary tasks, such as editing and generation of semantic-level scene objects.

References

[1] Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. Deep vit features as dense visual descriptors. *arXiv preprint arXiv:2112.05814*, 2021. 3

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 1, 2

[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. 1, 2, 6

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. *arXiv preprint arXiv:2304.06706*, 2023. 2

[5] 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. 2, 3, 4

[6] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG)*, 32(3), 2013. 2

[7] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, pages 333–350. Springer, 2022. 2

[8] 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*, 2023. 3

[9] Martin Eisemann, Bert De Decker, Marcus Magnor, Philippe Bekaert, Edilson De Aguiar, Naveed Ahmed, Christian Theobalt, and Anita Sellent. Floating textures. In *Computer graphics forum*, pages 409–418. Wiley Online Library, 2008. 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. 2

[11] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270, 2022. 4, 1

[12] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. 2

[13] 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. 2

[14] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M Seitz. Multi-view stereo for community photo collections. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007. 2

[15] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)*, 37(6):1–15, 2018. 2

[16] Tianyu Huang, Bowen Dong, Yunhan Yang, Xiaoshui Huang, Rynson WH Lau, Wanli Ouyang, and Wangmeng Zuo. Clip2point: Transfer clip to point cloud classification with image-depth pre-training. *arXiv preprint arXiv:2210.01055*, 2022. 2

[17] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 42(4):1–14, 2023. 1, 2, 3, 4, 6

[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, 2, 3, 6, 7, 4

[19] 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. 1, 2, 3, 6, 7, 4

[20] 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. 2

[21] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and Rene Ranftl. Language-driven semantic segmentation. In *International Conference on Learning Representations*, 2022. 6

[22] Kunhao Liu, Fangneng Zhan, Jiahui Zhang, Muyu Xu, Yingchen Yu, Abdulmotaleb El Saddik, Christian Theobalt, Eric Xing, and Shijian Lu. 3d open-vocabulary segmentation with foundation models. *arXiv preprint arXiv:2305.14093*, 2023. 2, 3, 6, 7, 4

[23] 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. *arXiv preprint arXiv:2305.14093*, 2023. 2

[24] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020. 2

[25] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. 6

[26] Yuheng Lu, Chenfeng Xu, Xiaobao Wei, Xiaodong Xie, Masayoshi Tomizuka, Kurt Keutzer, and Shanghang Zhang. Open-vocabulary point-cloud object detection without 3d annotation. 2023. 3

[27] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021. 2

[28] 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, 2, 5

[29] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 2

[30] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017. 6

[31] Songyou Peng, Kyle Genova, Chiyu "Max" Jiang, Andrea Tagliasacchi, Marc Pollefeys, and Thomas Funkhouser. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

[32] 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. 2, 3, 4

[33] Konstantinos Rematas, Andrew Liu, Pratul P. Srinivasan, Jonathan T. Barron, Andrea Tagliasacchi, Tom Funkhouser, and Vittorio Ferrari. Urban radiance fields. *CVPR*, 2022. 2

[34] Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulò, 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. 2

[35] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pages 835–846. 2006. 2

[36] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 2

[37] Ayça Takmaz, Elisabetta Fedele, Robert W. Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. OpenMask3D: Open-Vocabulary 3D Instance Segmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 3

[38] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8248–8258, 2022. 2

[39] 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. 2

[40] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 4

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

[42] Yuang Wang, Xingyi He, Sida Peng, Haotong Lin, Hujun Bao, and Xiaowei Zhou. Autorecon: Automated 3d object discovery and reconstruction. In *CVPR*, 2023. 3

[43] Jianzong Wu, Xiangtai Li, Shilin Xu, Haobo Yuan, Henghui Ding, Yibo Yang, Xia Li, Jiangning Zhang, Yunhai Tong, Xudong Jiang, Bernard Ghanem, and Dacheng Tao. Towards open vocabulary learning: A survey. *arXiv pre-print*, 2023. 3

[44] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022. 2

[45] Le Xue, Mingfei Gao, Chen Xing, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip: Learning unified representation of language, image and point cloud for 3d understanding. *arXiv preprint arXiv:2212.05171*, 2022. 2

[46] Le Xue, Ning Yu, Shu Zhang, Junnan Li, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip-2: Towards scalable multimodal pre-training for 3d understanding, 2023. 2

[47] Jianing Yang, Xuweiyi Chen, Shengyi Qian, David Fouhey, and Joyce Chai. Chat with nerf: Grounding 3d objects in neural radiance field through dialog, 2023. 2

[48] Jianing Yang, Xuweiyi Chen, Shengyi Qian, Nikhil Madaan, Madhavan Iyengar, David F. Fouhey, and Joyce Chai. Llm-grounder: Open-vocabulary 3d visual grounding with large language model as an agent, 2023. 2

[49] Junbo Zhang, Runpei Dong, and Kaisheng Ma. Clip-fo3d: Learning free open-world 3d scene representations from 2d dense clip, 2023. 3

[50] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 1, 2

[51] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018. 6

[52] Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. Pointclip: Point cloud understanding by clip. *arXiv preprint arXiv:2112.02413*, 2021. 2

[53] 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. 2

[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. 2

[55] Xiangyang Zhu, Renrui Zhang, Bowei He, Ziyu Guo, Ziyao Zeng, Zipeng Qin, Shanghang Zhang, and Peng Gao. Pointclip v2: Prompting clip and gpt for powerful 3d open-world learning. *arXiv preprint arXiv:2211.11682*, 2022. 2

Language Embedded 3D Gaussians for Open-Vocabulary Scene Understanding

Supplementary Material

We provide more details in this supplementary document, including load balancing loss (Sec. 7), inference strategy (Sec. 8), implementation details (Sec. 9), dataset details (Sec. 10) and more Results (Sec. 11).

7. Load Balancing Loss

To maximize utilization of the optimized feature space and avert quantization collapse, we introduce a load balancing loss, inspired by Switch Transformer [11]. When quantizing K features, the utilization ratio of each feature in $\mathcal{S}$ is calculated as:

$$m_i = \text{argmax}_j(\mathcal{D}(\mathbf{F}_i, \mathbf{f}_j)), \text{ where } \mathbf{f}_j \in \mathcal{S}, \tag{16}$$

$$\mathbf{r} = \frac{\sum_i^K \text{onehot}(m_i)}{K}, \tag{17}$$

where $\mathbf{r} \in \mathbb{R}^N$. We compute the mean selection probability for each feature over K quantizations:

$$\mathcal{D}(\mathbf{F}_i, \mathcal{S}) = [\mathcal{D}(\mathbf{F}_i, \mathbf{f}_1), \mathcal{D}(\mathbf{F}_i, \mathbf{f}_2), \ldots, \mathcal{D}(\mathbf{F}_i, \mathbf{f}_N)], \tag{18}$$

$$\mathbf{p} = \frac{\sum_i^K \text{Softmax}(\mathcal{D}(\mathbf{F}_i, \mathcal{S}))}{K}, \tag{19}$$

where $\mathcal{D}(\mathbf{F}_i, \mathcal{S}) \in \mathbb{R}^N$ and $\mathbf{p} \in \mathbb{R}^N$. The load balancing loss is then computed by the element-wise multiplication of $\mathbf{r}$ and $\mathbf{p}$, followed by their aggregation:

$$\mathcal{L}_{lb} = \sum^N (\mathbf{r} \circ \mathbf{p}), \tag{20}$$

where $\circ$ denotes the element-wise product.

8. Inference Strategy

In the inference stage, rasterization and alpha blending are employed to project the compact semantic features of 3D Gaussians into a 2D feature map. This feature map is then converted into the distribution of semantic indices using a trained MLP decoder and softmax activation, expressed as:

$$\mathcal{M}_{\text{infer}} = \text{Softmax}(D_{\text{MLP}}(R_s(\mathcal{G}; p_{\text{cam}}))), \tag{21}$$

where $R_s(\mathcal{G}; p_{\text{cam}}) \in \mathbb{R}^{H \times W \times d_s}$ denotes the rendered semantic features from a set of 3D Gaussians $\mathcal{G}$, as observed from the camera pose p_{cam}. Here, D_{MLP} symbolizes the trained MLP decoder of semantic features on 3D Gaussians. The result is a language feature index distribution $\mathcal{M}_{\text{infer}} \in \mathbb{R}^{H \times W \times N}$, where H and W represent the image's height and width, respectively. We finally acquire the language feature map, by multiplying $\mathcal{M}_{\text{infer}}$ with the quantized language features matrix $\mathbf{S} \in \mathbb{R}^{N \times d}$:

$$\mathcal{F} = \mathcal{M}_{\text{infer}}\mathbf{S}, \tag{22}$$

where $\mathcal{F} \in \mathbb{R}^{H \times W \times d}$ denotes the language feature map derived from the 3D Gaussians $\mathcal{G}$ observed from the camera pose p_{cam}. Figure 7 depicts the inference pipeline for generating language feature maps from language-embedded 3D Gaussians.

Utilizing the provided text prompt, we identify objects within the 3D scene by computing the relevance map of $\mathcal{F}$ in accordance with LERF [18].

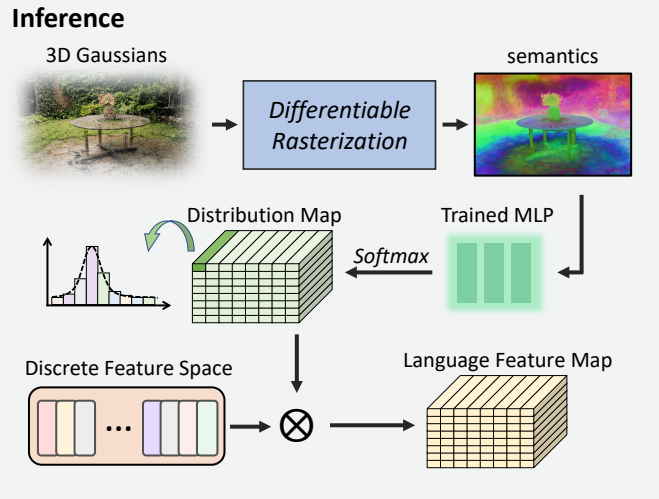

Figure 7. The inference pipeline for language feature maps from optimized 3D Gaussians. Each element in the distribution map $\mathcal{M}_{\text{infer}}$ represents a probability distribution of features in the Discrete Feature Space $\mathcal{S}$. We compute the corresponding language feature map $\mathcal{F}$ based on $\mathcal{S}$ utilizing these distributions.

	Layer	Config	Out Size
Input	-	-	$8 \times H \times W$
C1	Conv+ReLU	$128 \times 1 \times 1 / 1$	$128 \times H \times W$
C2	Conv+ReLU	$256 \times 1 \times 1 / 1$	$256 \times H \times W$
C3	Conv	$N \times 1 \times 1 / 1$	$N \times H \times W$

Table 3. Details of our semantic feature decoder D_{MLP}. In a layer characterized by $c \times w \times w/s$, c represents the number of filters, $w \times w$ indicates the filter size, and s denotes the stride size. The output dimensionality is expressed in terms of channel $\times$ height $\times$ width. The dimension of semantic feature on 3D Gaussians is 8 and N represents the size of discrete language feature space $\mathcal{S}$.

	Layer	Config	Out Size
PE	Positional Encoding	0	3
F1	Full-connected+ReLU	128	128
F2	Full-connected+ReLU	128	128
F3	Full-connected+ReLU	128	128
F4	Full-connected	8	8

Table 4. Details of the PE and MLP in the adaptive spatial smoothing. The input is the position of 3D Gaussian and the output is smoothed semantic feature s_{MLP}. The configure of PE layer represents the frequency of positional encoding.

9. Implementation Details

In Tab. 3 and Tab. 4, we present the implementation details of the semantic feature decoder D_{MLP} and the PE and MLP in the adaptive spatial smoothing, respectively. Furthermore, for the size N of the discrete language feature space $\mathcal{S}$, we set $N = 32$ for the "kitchen" scene, $N = 64$ for the "bonsai" scene, and $N = 128$ for other scenes. N, as a hyperparameter, controls the capacity of semantic information in the discrete language feature space $\mathcal{S}$ and can be adjusted according to the richness of semantic information in the scene.

10. Datasets

To concurrently assess the quality of visual and semantic embeddings, six scenes from the Mip-NeRF360 dataset [3] are chosen for quantitative and qualitative evaluation. The 'Stump' scene is excluded due to its insufficient semantic content. The evaluation set of each scene is manually annotated with segmentation maps, which are created for the primary objects in each scene. The text prompts corresponding to these annotated objects are listed in Tab. 5. Additionally, segmentation masks for some objects in our dataset are illustrated in Fig. 8.

11. More Results

Further qualitative results are presented to illustrate the comparison of visual quality (Fig. 10), the evaluation of novel view synthesis and query accuracy (Fig. 9), and the exploration of open-vocabulary queries (Fig. 11).

Scene	Positive Words
bicycle	green grass, white bicycle, tire, bench, asphalt ground, silver oak tree
bonsai	piano keyboard, bicycle, purple table cloth, black stool, plastic bonsai tree, dark grey patterned carpet
counter	jar of coconut oil, fruit oranges, onions, plants, blue oven gloves, wood rolling pin, free range eggs box, stable bread, garofalo pasta, napolina tomatoes, gold ripple baking pan
garden	football, wood round table, green grass, wood pot, elderflower, green plant, bricks wall, windows, stone ground
kitchen	LEGO Technic 856 Bulldozer, basket weave cloth, wood plat, old pink striped cloth, red oven gloves
room	blue grey chair, curtain, brown shoes, books, windows, door, piano keyboard, wood floor, wine glasses and bottles, yucca plant, deep green carpets

Table 5. Text prompts used for evaluating quality and accuracy of open-vocabulary query.

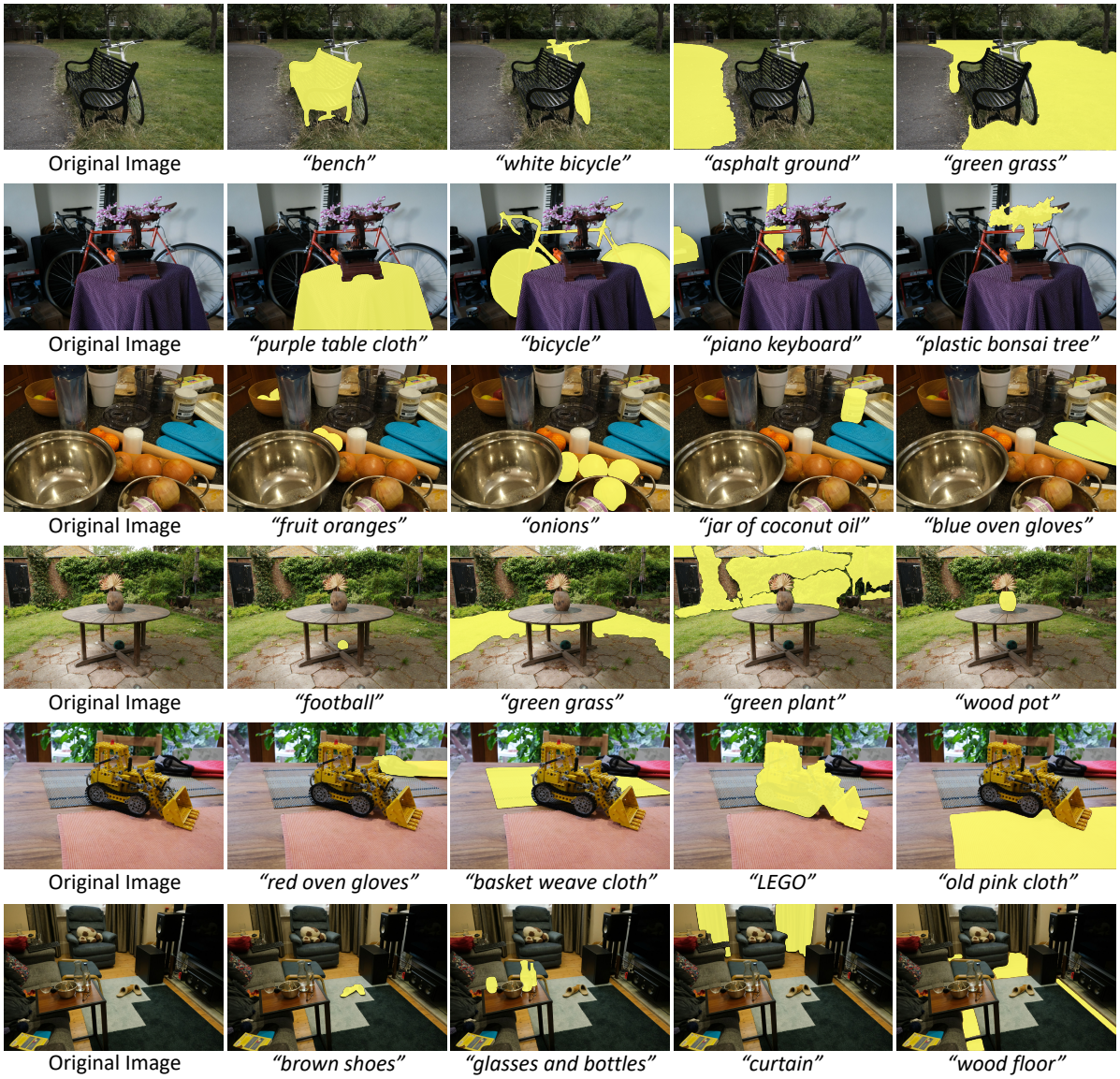

Figure 8. Ground truth segmentation masks for some objects in our dataset. In each scene, we select primary, unambiguous objects for semantic annotation. This includes both large and small objects, as well as challenging entities with complex geometric structures or transparent and translucent properties, such as bicycles, windows, and water glasses.

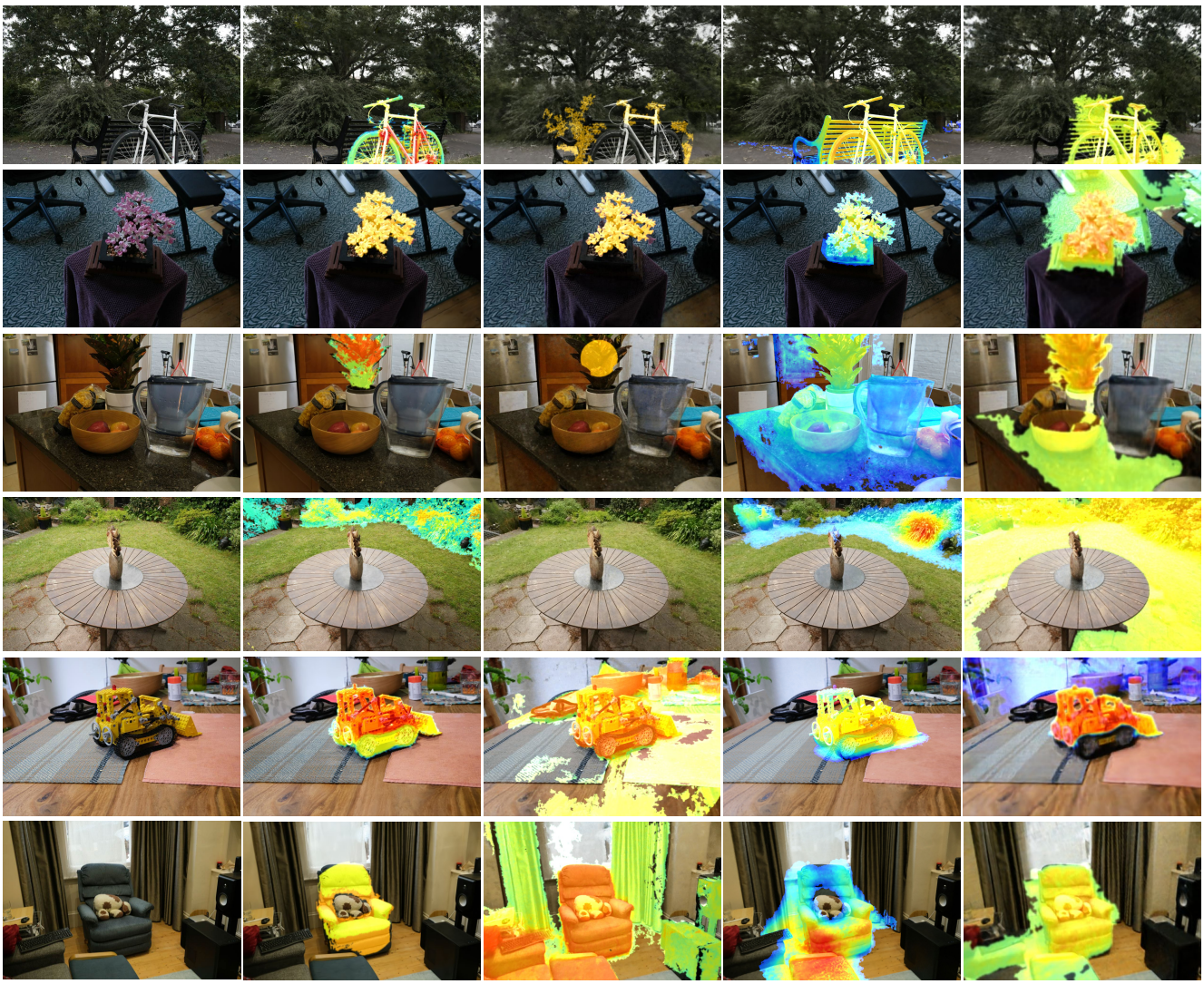

Figure 9. Comparison of novel view synthesis quality and open-vocabulary query accuracy. Left to right: Ground truth novel view synthesis, novel view images with relevance visualization from our method, DFF [19], LeRF [18], and 3DOVS [22]. Top to bottom: Query words "white bicycle", "bonsai", "plants", "green plant", "LEGO Technic 856 Bulldozer", and "blue grey chair".

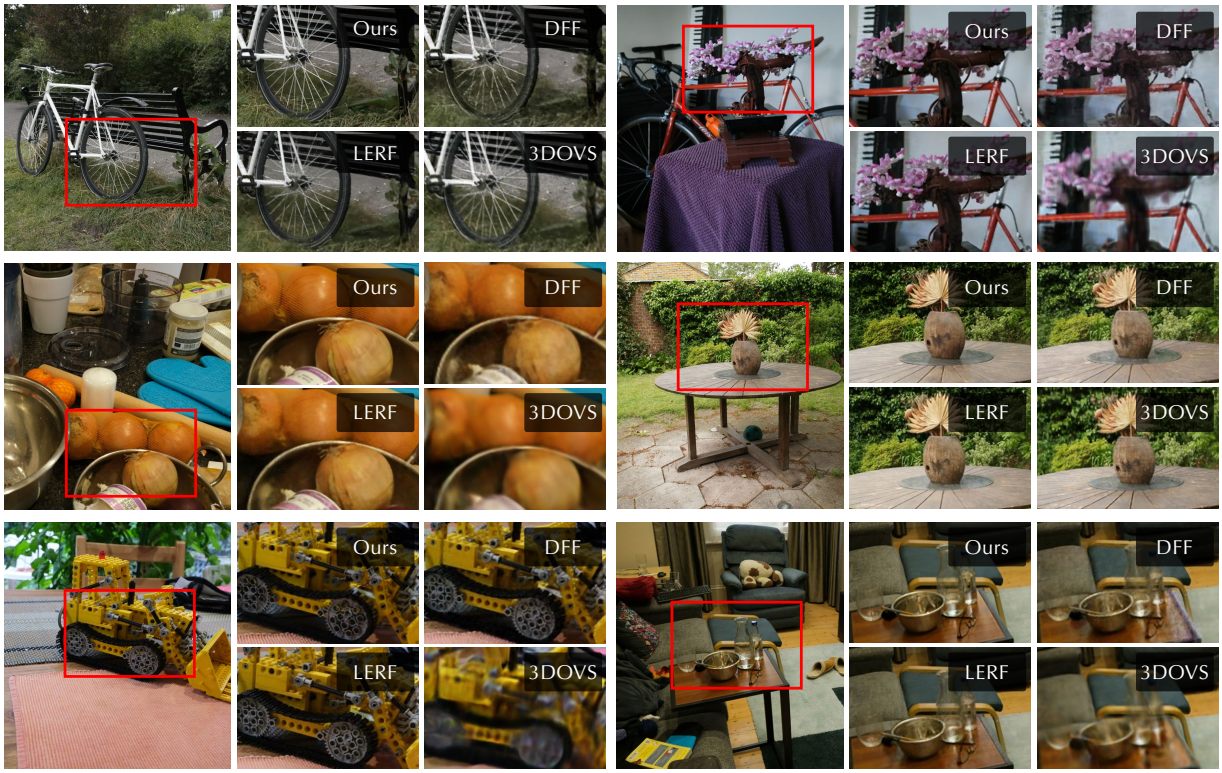

Figure 10. Comparison of the quality of novel view synthesis. Even with dense language features embedded into the 3D Gaussians, our method still only requires a reasonable amount of memory, thus allowing a massive amount of points to be rendered and optimized at the same time, achieving the best visual quality with more details compared to other methods.

Figure 11. Examples of various open-vocabulary queries. Our approach enables accurate open-vocabulary queries using a diverse class of word types, including but not limited to, visual attributes, general terms, materials, olfactory properties, and related actions.