Dr. Splat: Directly Referring 3D Gaussian Splatting via Direct Language Embedding Registration

Kim Jun-Seong[1] GeonU Kim[1] Kim Yu-Ji[1] Yu-Chiang Frank Wang[2]
Jaesung Choe[2*] Tae-Hyun Oh[3*]

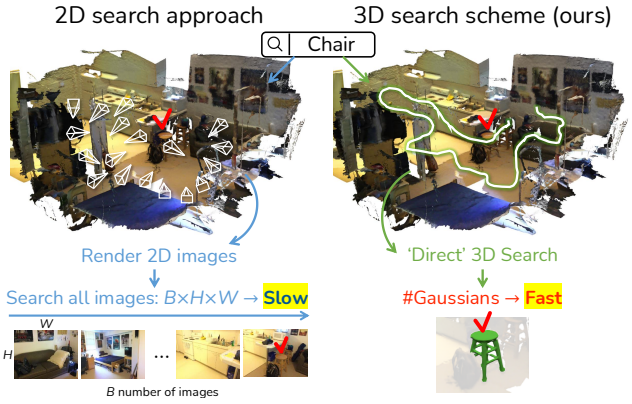
[1]POSTECH [2]NVIDIA [3]KAIST

Abstract

We introduce Dr. Splat, a novel approach for open-vocabulary 3D scene understanding leveraging 3D Gaussian Splatting. Unlike existing language-embedded 3DGS methods, which rely on a rendering process, our method directly associates language-aligned CLIP embeddings with 3D Gaussians for holistic 3D scene understanding. The key of our method is a language feature registration technique where CLIP embeddings are assigned to the dominant Gaussians intersected by each pixel-ray. Moreover, we integrate Product Quantization (PQ) trained on general large-scale image data to compactly represent embeddings without per-scene optimization. Experiments demonstrate that our approach significantly outperforms existing approaches in 3D perception benchmarks, such as open-vocabulary 3D semantic segmentation, 3D object localization, and 3D object selection tasks. For video results, please visit : https://drsplat.github.io/

1. Introduction

Open-vocabulary 3D scene understanding represents a significant challenge in the field of computer vision, with applications spanning autonomous navigation, robotics, and augmented reality. This approach aims to enable the interpretation and referencing of 3D spatial information through natural language, allowing for applicability beyond a restricted set of predefined categories [2, 3, 26, 27, 31, 34, 38]. Previously, open-vocabulary 3D scene understanding has been explored using point-cloud-based methods [11, 15, 17, 25, 28, 33, 36]. Recently, the 3D Gaussian Splatting (3DGS) [16] has introduced a continuous representation integrated on explicit 3D Gaussians, which differs from traditional point-cloud approaches, enabling rapid progress in practical applications [39]. Current research has begun to explore methods for associating language-based features with 3D Gaussian splats to enhance scene understanding capabilities.



	Search domain	Per-scene opt.	Feature distill.	Search	DB size
LERF [17]	2D	required	$\sim 24h$	slow	large
LangSplat [28]	2D	required	$\sim 4h$	slow	large
LEGaussians [32]	2D	required	$\sim 4h$	slow	large
OpenGaussian [35]	3D	required	$\sim 1h$	fast	small
Dr. Splat (Ours)	3D	none	$\sim 10m$	fast	small

Figure 1. Comparison of 2D (left) vs. our direct 3D search (right) for open-vocabulary 3D scene understanding. The 2D approach relies on multiview rendering, incurring high computational costs. Our method directly links language features to 3D Gaussians, enabling efficient and complete spatial coverage. The table highlights Dr. Splat 's superior efficiency over related methods.

Several recent approaches [28, 32, 41] introduce 3D Gaussian representation [16] into the open-vocabulary scene understanding. This unique representation uses 3D Gaussians to achieve high-quality scene rendering, offering a more structured representation that addresses some limitations of point clouds. Building on this, these methods employ 2D vision-language models to transfer language knowledge to 3D Gaussians "via rendered feature maps".

Despite its promise, such rendering-based distillation methods [28, 32] share two limitations. First, we found that there is a discrepancy between optimized embeddings in 3D Gaussians and 2D language-aligned embeddings. This gap arises mainly from an intermediate rendering step that may distort CLIP embeddings during training. Then, the reliance on rendering impedes holistic 3D scene understanding, addi-

*Corresponding Authors

tional task-processing such as 3D semantic segmentation and 3D object localization, and making full spatial coverage calculations less efficient than direct 3D Gaussian methods [35] including ours as illustrated in Fig. 1.

To address this issue, this work proposes Dr. Splat. Our method bypasses the rendering stage, enabling direct interaction with 3D Gaussians for registering and referring the well-preserved language-aligned CLIP embeddings in the 3D space. This makes our Dr. Splat clearly distinguishable from prior works, facilitating a seamless integration of representative embeddings from 2D vision language models into the 3D spatial structure without compromising exhaustive rendering process that has been exploited [14, 28, 32, 39–41]. Moreover, we propose to use a Product Quantization (PQ) feature encoding method to represent embeddings compactly and efficiently without any per-scene optimization. Rather than storing full-length feature vectors or per-scene specifically compressed embeddings [14, 28, 32, 39–41], each Gaussian in our Dr. Splat stores an index from a pre-trained PQ, significantly reducing memory usage up to 6.25% compression ratio. By preserving the richness of embeddings while reducing memory usage, PQ is integral to our framework's high scalability and its ability to perform 3D perception tasks, such as open-vocabulary 3D object localization, 3D object selection, and 3D semantic segmentation. Our contributions are summarized as follows:

- We propose Dr. Splat, direct registration and referencing of language-aligned features in 3D Gaussians, bypassing intermediate rendering and preserving feature accuracy.
- We introduce the PQ encoding method for compact feature representation, reducing memory usage while maintaining essential 3D feature properties.
- We present a novel evaluation protocol to assess accuracy of 3D localization and segmentation for 3D Gaussians, with pseudo-labeling methods and volume-aware metrics.

2. Related Work and Motivation

Language-based 3D scene understanding. Open-set 3D scene understanding has seen considerable advancements, with a focus on methods that leverage language knowledge into 3D representation such as point clouds, neural radiance fields (NeRF) [24], and Gaussian Splatting [16] for 3D comprehension. Point-based methods [5, 12, 15, 23, 25, 36, 37] in open-vocabulary settings process point cloud data trained from language embeddings [21, 29] for open-set categories.

NeRF-based approaches [6, 17, 19, 22, 30] leverage semantic embeddings from 2D foundation models, such as CLIP [29], LSeg [21] and DINO [1] for open-vocabulary understanding. While the rendering process enhances 2D perception tasks, the implicit nature of NeRF constrains the holistic understanding of 3D structures and dominantly provides 'rendered' feature maps.

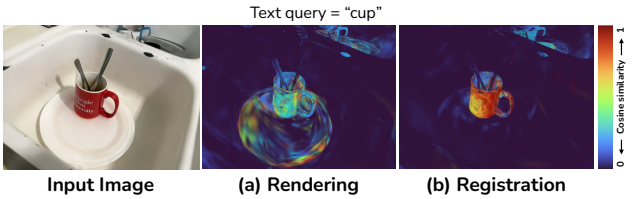


Figure 2. Visualization of discrepancy in rendered 2D features and 3D features. Color indicates a cosine similarity score between query features from a text query and either (a) 3D features distilled by 2D rendering [28] or (b) directly registered 3D features.

3D Gaussian Splatting (3DGS) [16] has emerged as a promising rendering method, as well as a novel representation for open-vocabulary 3D scene understanding. Since this research is the close related work with our work, we first elucidate the preliminary of 3DGS, followed by focusing on language embedded 3DGS as follows.

Preliminary of 3D Gaussian Splatting. 3DGS [16] encodes appearance and geometry of the target scene into the 3D Gaussian representation. Each 3D primitive representation is expressed as a 3D Gaussian distribution having mean $\boldsymbol{\mu} = [x_\mu, y_\mu, z_\mu]^\top$ for 3D position and covariance matrix $\Sigma_{3D} \in \mathbb{R}^{3\times3}$ for 3D volume, as well as the opacity value α and the color $\mathbf{c}$. In particular, the covariance matrix is decomposed into the scale matrix $S \in \mathbb{R}^{3\times3}$ and the rotation matrix $R \in SO(3)$, $\Sigma_{3D} = RSS^\top R^\top$. In brief, N numbers of 3D Gaussians can be parametrized as $\Theta = \{\boldsymbol{\theta}_i\}_{i=1}^N = \{\boldsymbol{\mu}_i, S_i, R_i, \alpha_i, \mathbf{c}_i\}_{i=1}^N$. 3D Gaussians Θ are used to render a 2D pixel color $\hat{\mathbf{c}}$ computed as:

$$\hat{\mathbf{c}}(\theta) = \sum_{i=1}^{N} T_i \tilde{\alpha}_i \mathbf{c}_i, \text{ s.t. } \tilde{\alpha}_i = \alpha_i \exp\left(-\tfrac{1}{2}\mathbf{d}^\top \Sigma_{2D}^{-1}\mathbf{d}\right), \quad (1)$$

T_i is a transmittance, $\tilde{\alpha}_i$ is an effective opacity value computed from the Gaussian's opacity α, the pixel distance $\mathbf{d} \in \mathbb{R}^{2\times1}$ from the target pixel to the projected center location of the Gaussian in pixel, and Σ_{2D} is the 2D covariance matrix in the image domain obtained from the splatting algorithm [16, 42]. The 3D Gaussian parameters Θ of a scene are optimized by minimizing the rendering loss between the input image color $\mathbf{c}$ and the rendered color $\hat{\mathbf{c}}(\theta)$ in Eq. (1) as $\arg\min_\theta \|\mathbf{c} - \hat{\mathbf{c}}(\theta)\|_F^2$.

Language embedded 3D Gaussian Splatting. The basic idea of the language embedded Gaussian representation [9, 14, 20, 28, 32, 39–41] is to replace the color rendering to language embedding rendering. Language embedded 3D Gaussians are parameterized as $\Phi = \{\theta_i, \tilde{\mathbf{f}}_i\}_{i=1}^N = \{\boldsymbol{\mu}_i, S_i, R_i, \alpha_i, \mathbf{c}_i, \tilde{\mathbf{f}}_i\}_{i=1}^N$, where $\tilde{\mathbf{f}}_i$ denotes Gaussian-registered language embeddings across N numbers 3D Gaussians which will be discussed soon. Then, analogous to the color rendering Eq. (1), the language embedding rendering is expressed as:

$$\hat{\mathbf{f}} = \sum_{i=1}^{N} T_i \tilde{\alpha}_i \tilde{\mathbf{f}}_i, \quad (2)$$

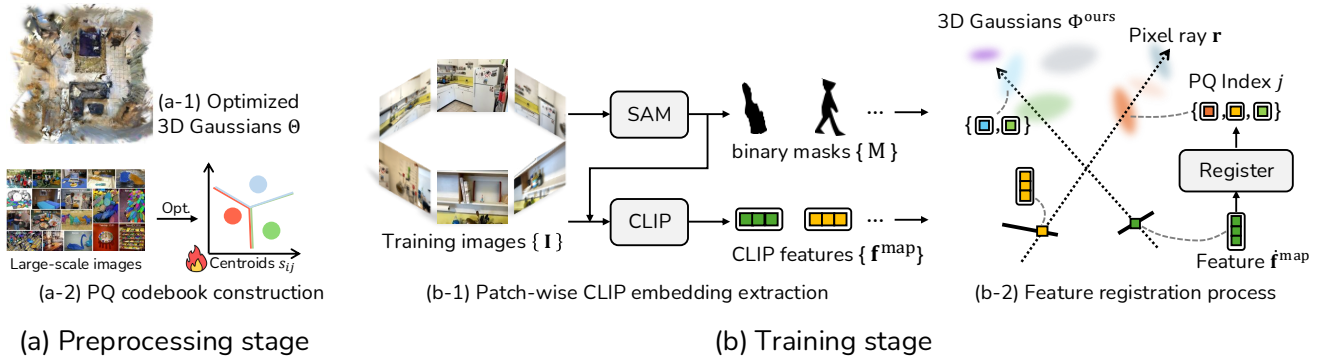


Figure 3. Overview of Dr. Splat. (a) In the preprocessing stage, we compute optimized 3D Gaussians [16] and Product Quantization (PQ) codebook construction. (b) During training, we extract CLIP embeddings from given images $\{\mathbf{I}\}$, and then proceed feature registration process (Sec. 3.1). Finally, we obtain 3D Gaussians Φ^{ours} with PQ indices $\{j\}$ (Sec. 3.2).

where $\hat{\mathbf{f}}$ denotes a rendered language embedding. Likewise, the Gaussian-registered language embeddings $\{\tilde{\mathbf{f}}\}$ are optimized by minimizing the rendering loss between the 2D language embedding $\mathbf{f}$ extracted from an input image and a rendered language embedding map $\hat{\mathbf{f}}$ as $\arg\min_{\{\tilde{\mathbf{f}}\}} \|\mathbf{f}-\hat{\mathbf{f}}\|_F^2$ at each corresponding pixel. This can be regarded as distilling vision language models into Gaussian-registered language embedding $\tilde{\mathbf{f}}$ through volume rendering Eq. (2). The Gaussian-registered language embeddings are separately trained after pre-training and fixing the pre-trained 3DGS Θ for a scene. The language embeddings to be distilled are typically obtained from CLIP [29]. Since storing 32-bit 512-D CLIP features $\mathbf{f}$ in every 3D Gaussians is memory-expensive, one can use a compressed feature per scene depending on the needs [14, 32, 39–41].

Motivation. Such language-embedded radiance fields provide useful representation and language interfaces for many practical and crucial applications. While most of existing works focus on the training efficiency, the complexity in inference time has barely been discussed. Considering a scenario to text-query a 3D location of the language-embedded Gaussians, *i.e.*, 3D localization, the aforementioned methods first require rendering a 2D language embedding map at each specific camera pose. We cannot directly retrieve over the distributed embeddings $\{\tilde{\mathbf{f}}_i\}$ in 3D Gaussians, because the embeddings do not carry language information directly, but their weighted summed (rendered) features $\hat{\mathbf{f}}$ do. This issue becomes even severer with compressed features as in [28]: their decompression decoders are not designed for and incompatible with directly applying to the distributed compressed language embeddings in each 3D Gaussian, yielding degenerated CLIP decoding (refer to Fig. 2).

This introduces multiple challenges and hassles. First, it is challenging to find the best or proper camera rendering views that contain the object to find. One may attempt to pre-compute the minimal number of cameras and their camera poses that cover all the 3D Gaussians in a scene with proper resolutions, similarly by point-based approach [11]. However, this is a well-known set covering problem [7] with constraints which is known to be an NP-hard problem.

Second, even with pre-computed rendered views, the retrieval complexity over the rendered images remains substantial [8]. Suppose a scene consisting of one million Gaussians, but just a *single* rendered language embedding map in pixel domain already has nearly a million pixels; thus, we need a dedicated system to efficiently retrieve over all the views. Third, since the retrieval is conducted in the 2D space, to find a 3D location, we need a separate mechanism to lift the localization to the 3D space, *i.e.*, increasing the system complexity. In addition, 32-bit floating 512-Dimension CLIP features for millions of Gaussian are memory intensive, which is often not manageable. To reduce this burden, the existing methods [35] apply compressions with per-scene optimized codebooks, which hinders extension or generalization to other scenes.

To overcome these, we propose a training-free algorithm for the direct allocation of language embeddings to 3D Gaussians, allowing efficient computation and interaction within the 3D space. As a concurrent work, OpenGaussian [35] tackles a similar challenge with our work, but still requires per-scene codebook construction Fig. 1.

3. Dr. Splat

This section provides details of our method. We first explain how we directly register CLIP embeddings into Gaussian-registered language embeddings, Sec. 3.1. Then, we introduce Product Quantization (PQ) into our framework to efficiently store Gaussian-registered language embeddings, Sec. 3.2. Lastly, we describe the inference stage for text query-based 3D Gaussian localization, Sec. 3.3.

3.1. Feature registration process

Our goal is to reconstruct a language embedded 3D space represented by 3D Gaussians Φ, which we can directly inter-

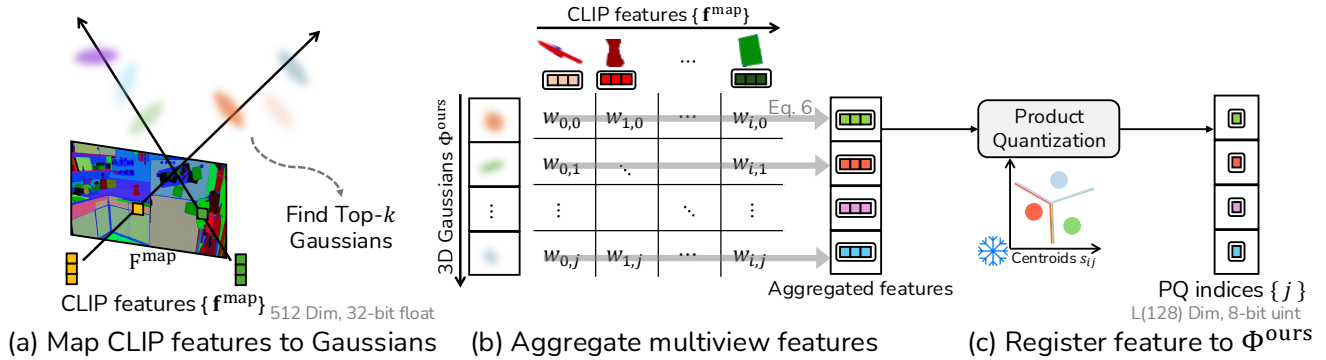


Figure 4. Feature registration process in Dr. Splat. (a) We first map per-pixel CLIP embeddings $\{\mathbf{f}^{\text{map}}\}$ to Gaussians. Here, we only map dominant k Gaussians along pixel ray r, named Top-k Gaussians. (b) After collecting embeddings, we compute aggregated features (Eq. (6)). (c) Finally, we re-use PQ to obtain the PQ indices j of aggregated features and update Gaussian parameters Φ^{ours}.

act in 3D space without feature rendering Eq. (2). For that, following LangSplat [28], we begin by extracting per-pixel CLIP embedding maps $\mathbf{F}^{\text{map}} \in \mathbb{R}^{D\times H\times W}$ from training images of the target scenes, where D is the dimension of CLIP embeddings, H and W are the height and width of the training images. Given training images, we extracts a dictionary of binary masks and language embeddings extracted from the images as: $\mathcal{F}^{\text{map}} = \{\mathbf{M}_j : \mathbf{f}_j^{\text{map}} \mid j = 1, ..., M\}$, where $\mathbf{M}_j \in \mathbb{R}^{H\times W}$ is a binary mask extracted using SAM [18] and $\mathbf{f}_j^{\text{map}} \in \mathbb{R}^D$ is a corresponding CLIP embedding from a cropped image with $\mathbf{M}_j$. Each mask $\mathbf{M}_j$ belongs to an image, and the masks are not overlapped to each other. With this dictionary, a CLIP embedding map $\mathbf{F}^{\text{map}}(\mathbf{I}, \mathbf{r})$ at a pixel $\mathbf{r}$ in a training image $\mathbf{I}$ is computed as:

$$\mathbf{F}^{\text{map}}(\mathbf{I}, \mathbf{r}) = \sum_{j=1}^{M} \mathbf{M}_j(\mathbf{I}, \mathbf{r}) \cdot \mathbf{f}_j^{\text{map}}, \tag{3}$$

where $\mathbf{M}_j(\mathbf{I}, \mathbf{r}) \in \{0, 1\}$ indicates whether the mask $\mathbf{M}_j$ contains the pixel $\mathbf{r}$ in the image $\mathbf{I}$. Using $\mathbf{F}^{\text{map}}$, we reconstruct language embedded 3D Gaussians via a novel feature registration process as visualized in Fig. 3.

During the feature registration process, our algorithm iterates through training images of the scene. Using projection relation, we link 3D Gaussians Φ to CLIP embeddings. Each Gaussian can link to multiple CLIP embeddings derived from different images. Then we aggregate collected embeddings to a single embedding to be assigned to each Gaussian. To ensure a consistent aggregation of the embeddings from multi-view images, we first compute a weight $w_i(\mathbf{I}, \mathbf{r})$ representing the contribution of θ_i to construct each pixel $\mathbf{r}$ in a training image $\mathbf{I}$. The weights are computed with the volume rendering equation Eq. (1) as:

$$w_i(\mathbf{I}, \mathbf{r}) = T_i(\mathbf{I}, \mathbf{r}) \cdot \tilde{\alpha}_i(\mathbf{I}, \mathbf{r}), \tag{4}$$

where $T_i(\mathbf{I}, \mathbf{r})$ and $\tilde{\alpha}_i(\mathbf{I}, \mathbf{r})$ are the transmittance and the effective opacity value of θ_i for a pixel $\mathbf{r}$ in an image $\mathbf{I}$, stated in Eq. (1). With the per-pixel weights, we calculate w_{ij} representing a weight between each Gaussian θ_i and corresponding language embedding maps $\mathbf{f}_j^{\text{map}}$, which is for aggregating CLIP embeddings from $\mathbf{F}^{\text{map}}$ and register the embedding to each Gaussian. The weights are computed as:

$$w_{ij} = \sum\nolimits_{\mathbf{I}\in\mathcal{I}} \sum\nolimits_{\mathbf{r}\in\mathbf{I}} \mathbf{M}_j(\mathbf{I}, \mathbf{r}) \cdot w_i(\mathbf{I}, \mathbf{r}), \tag{5}$$

where $\mathcal{I}$ is the set of the training images. In this iterative process, we aggregate weights only for Top-k Gaussians with the highest weights $w_i(\mathbf{I}, \mathbf{r})$, along the ray of each pixel ray $\mathbf{r}$ (see Fig. 4). After aggregation, we prune the Gaussians which are not assigned any weight, *i.e.*, $\sum_{j=1}^{M} w_{ij} = 0$. This summation aggregates weights between Gaussians and the CLIP embeddings by linking per-pixel weights $w_i(\mathbf{I}, \mathbf{r})$ of each Gaussian to its corresponding CLIP embeddings. With the obtained weights, we register an aggregated feature $\dot{\mathbf{f}}_i$ to each Gaussian with weighted-averaging as:

$$\dot{\mathbf{f}}_i = \mathbf{f}_i / ||\mathbf{f}_i||_2, \text{ where } \mathbf{f}_i = \sum_{j=1}^{M} \frac{w_{ij}}{\sum_{k=1}^{M} w_{ik}} \mathbf{f}_j^{\text{map}}. \tag{6}$$

This process enables 3D-aware feature registration to be consistent across various viewpoints, by aggregating features in the original high-dimensional feature space. The proposed process can be interpreted as an inverse volume rendering without gradient-based optimization, which enables our method to be faster than the prior methods requiring per-scene gradient-based optimization [25, 28, 32] for feature registration in 3D space.

3.2. Product-Quantized CLIP embeddings

Memory efficiency is a challenge in 3D scene representations, especially when associating Gaussians with high-dimensional feature vectors. LangSplat [28] addresses this by introducing an encoder-decoder network, while LeGaussian [32] and OpenGaussian [35] utilize codebook construction. However, these approaches introduce additional per-

Methods	mIoU					mAcc @ 0.25				
	waldo_kitchen	ramen	figurines	teatime	Mean	waldo_kitchen	ramen	figurines	teatime	Mean
LangSplat-m [28]	8.29	6.11	8.33	16.58	9.83	13.64	14.08	8.93	27.12	15.94
OpenGaussian [35]	34.60	23.87	**59.33**	54.44	43.06	50.00	**35.21**	80.36	72.88	59.61
Ours (Top-10)	37.05	24.33	54.42	**57.35**	43.29	**63.64**	**35.21**	80.36	**77.97**	64.30
Ours (Top-20)	38.33	24.58	53.94	56.19	43.26	**63.64**	**35.21**	**82.14**	76.27	**64.32**
Ours (Top-40)	**39.07**	**24.70**	53.36	57.20	**43.58**	**63.64**	**35.21**	80.36	76.27	63.87

Table 1. 3D object selection results on the LeRF-OVS dataset [17]. To measure 3D object selection performance, we calculate 2D segmentation accuracy on rendering of selected 3D Gaussians. Note that our model does not require per-scene optimization, demonstrating its robustness across diverse scenes. **Bold** and Underline stand for first and second best performance.

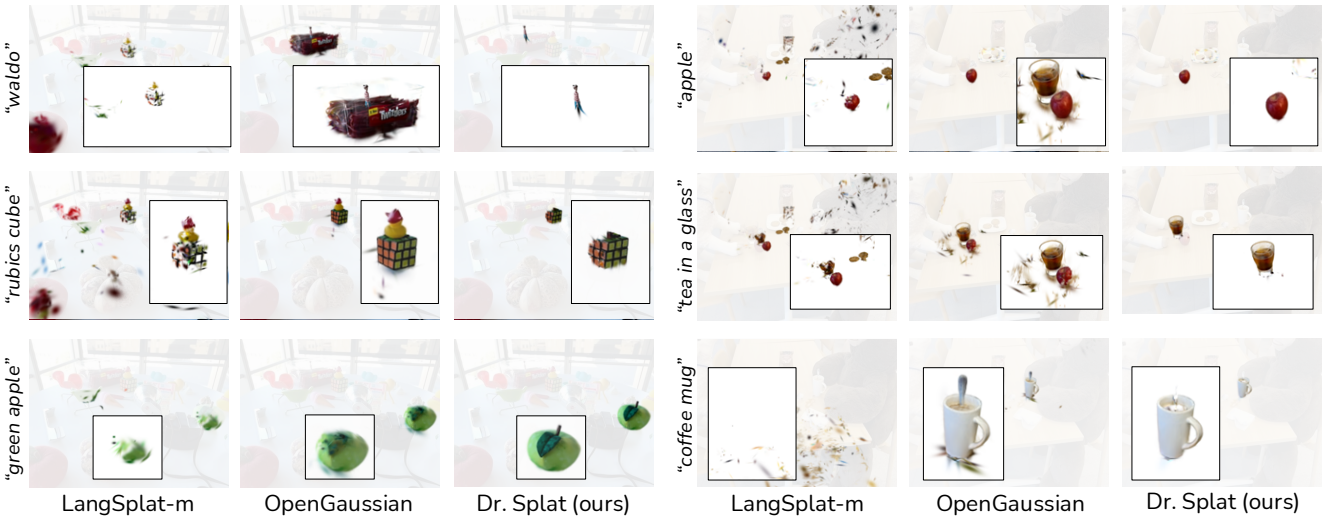


Figure 5. Qualitative results of the object selection on the LeRF-OVS dataset [17]. We visualize rendering of selected 3D Gaussians for LangSplat [28], OpenGaussian [35], and ours. For LangSplat, activations are often distributed randomly, fail to localize the target. OpenGaussian often struggles to distinguish closely situated objects. In contrast, our model shows activations precisely limited to the queried object regions, effectively localizing only the relevant areas.

scene computational costs for scene-specific parameter tuning of neural networks or codebooks (see Fig. 1). In contrast, we propose to use Product Quantization (PQ) on a large-scale image dataset, eliminating per-scene training.

Product Quantization. PQ [13] is a widely used technique for efficient embedding compression, particularly valuable in large-scale applications. The PQ process begins by dividing the original D-dimensional feature vector $\mathbf{v}$ into L sub-vectors: $\mathbf{v} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_L]$. Each sub-vector $\mathbf{v}_i$ is then independently quantized to a predefined number of centroids s_{ij} in a predefined codebook S_i for that sub-vector. These centroids are learned via clustering, creating a codebook for each subspace. Once the centroids are established, each sub-vector is replaced by the index of the nearest centroid in its respective codebook. The centroid indices $j_i = [j_{i1}, j_{i2}, \ldots, j_{iL}]$ are optimized by minimizing $\arg\min_k \|\mathbf{v}_i - s_{ik}\|$ to quantize a given vector $\mathbf{v}_i$ where j_{ik} is an 8-bit unsigned integer.

Then, we can measure the distance between the query and data by adding distances between coarse centroids. Once the distances between centroids are computed as a lookup table, the computation shifts to simple indexing, which reduces the search complexity from $\mathcal{O}(D)$ to $\mathcal{O}(1)$ for a D dimension sample. This approach notably reduces computational complexity, making it suitable for large-scale search.

In our setup for language-based 3D scene understanding, we build PQ centroids based on CLIP embeddings using a large-scale image dataset, the LVIS dataset [10], that contains over 1.2M instances covering various long-tail classes and ground truth segmentation. We extract instance patches from images and collect patch-wise CLIP embeddings. After we build this CLIP embedding database, we proceed with the construction of the centroid codebook for our PQ. Once PQ is trained, any query embedding can be approximated by assigning the closest centroid for each subvector. This is a one-time procedure; once we determine the codebook, we can use it for any scene generally. In our setup, each embedding is represented as a sequence of centroid indices rather than a high-dimensional vector. Accordingly, our language embedded Gaussians are parametrized as $\Phi^{\text{ours}} = \{\phi_i^{\text{ours}}\}_{i=1}^N = \{\theta_i, j_i\}_{i=1}^N$. where the aggregated

	3D	19 classes		
	mIoU	IoU > 0.15	IoU > 0.3	IoU > 0.45
LangSplat-m [28]	8.0	17.1	7.8	2.9
LEGaussians-m [32]	9.5	19.1	8.9	7.3
OpenGaussian [35]	<u>25.2</u>	<u>59.5</u>	<u>38.0</u>	18.3
Ours (Top-20)	25.0	**60.7**	**40.3**	<u>20.0</u>
Ours (Top-40)	**25.4**	**60.7**	**40.3**	**25.6**

(a) 3D object localization task.

	19 classes		15 classes		10 classes	
	mIoU	mAcc.	mIoU	mAcc.	mIoU	mAcc.
LangSplat-m [28]	2.0	9.2	4.9	14.6	8.0	23.9
LEGaussians-m [32]	1.6	7.9	4.6	16.1	7.7	24.9
OpenGaussian [35]	**30.1**	<u>46.5</u>	<u>38.1</u>	<u>56.8</u>	<u>49.7</u>	<u>71.4</u>
Ours (Top-20)	28.0	44.6	**38.2**	**60.4**	47.2	68.9
Ours (Top-40)	<u>29.6</u>	**47.7**	**38.2**	**60.4**	**50.2**	**73.5**

(b) Open-vocabulary 3D semantic segmentation task.

Table 2. Quantitative comparison in the ScanNet dataset [4]. Left: Localization prediction is defined as 3D regions with a text similarity score above threshold. Right: We assign segmentation labels by finding max activations among all classes. Note that **Bold** and <u>Underline</u> stand for first and second best performance, respectively.

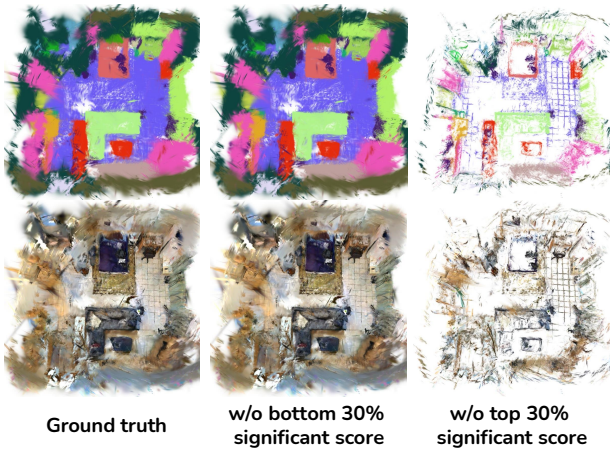


Figure 6. Limitations of point-based IoU measurement. This figure shows the effect of removing the top and bottom 30% of Gaussians according to the proposed significant score, implying that volume differences significantly impact 3D accuracy. The results highlight the need for the proposed IoU metric for 3D Gaussians.

feature $\dot{\mathbf{f}}_i$ are converted as a quantized feature $\bar{\mathbf{f}}_i$ by the corresponding PQ index j_i.

3.3. Text-query based 3D localization

After training 3D Gaussians Φ^{ours} with our feature registration process and PQ, we describe the details of an inference mode that facilitates direct interaction with 3DGS upon receiving input queries, such as text. This is related to similarity score computation between a query and sources, *i.e*. Gaussian embeddings. Given a text, we first extract a query feature $\mathbf{q}$ using CLIP text encoder [29]. We reconstruct the quantized features $\{\bar{\mathbf{f}}_i\}_{i=1}^{N}$ from the stored PQ indices $\{j_i\}_{i=1}^{N}$. Then, we compute a cosine similarity score between the query feature $\mathbf{q}$ and all quantized features.

Despite its simplicity, solely relying on the cosine similarity may result in diminished discriminability across certain similarity scores.

To address this limitation, we incorporate a re-ranking process based on relative activation with respect to the canonical feature. For this process, we adopt the relevancy scoring method proposed in LeRF [17], which enables more precise similarity analysis for a query. Specifically, each rendered language embedding, $\mathbf{f}^{\text{map}}$ and a text query feature $\mathbf{q}$, yield a relevance score determined by, $\min_i \frac{\exp(\mathbf{f}^{\text{map}}\cdot\mathbf{q})}{\exp(\mathbf{f}^{\text{map}}\cdot\mathbf{q})+\exp(\mathbf{f}^{\text{map}}\cdot\mathbf{f}^{\text{canon, i}})}$, where $(\cdot)$ is an element-wise dot product operator and $\mathbf{f}^{\text{canon,i}}$ indicates CLIP embeddings of a designated canonical term selected from a set of "object," "things," "stuff," and "texture". Then, we sample 3D Gaussians based on the relevance score for downstream tasks.

4. Experiments

Dataset. We use two datasets to evaluate the 3D scene understanding performance. For the 3D object selection task (Sec. 4.1), we use the LERF [17] dataset annotated by LangSplat [28], which consists of several multi-view images of 3D scenes containing long-tail objects and includes ground truth 2D ground truth annotations for texture queries. For 3D object localization Sec. 4.2 and 3D semantic segmentation Sec. 4.3 task, we employ the ScanNet [4] dataset. ScanNet is a large-scale benchmark that provides data on indoor scenes, including calibrated RGBD images and 3D point clouds with ground-truth semantic labels. We randomly select eight scenes from ScanNet for the experiments.

Competing methods. The only method available for a fair comparison with our method is the concurrent work, OpenGaussian [35]. To study the various aspects of our method, we introduce baseline methods modified from rasterization-based ones [28, 32], for direct 3D referring operation, denoted as LangSplat-m and LEGaussians-m. As discussed in Sec. 2, without modification, global search over a whole scene is quite demanding. To ensure fair evaluation, we use the same initial 3D Gaussians being trained only using RGB inputs for all comparing methods, and freeze the Gaussians during the language feature allocation process. Also, the per-pixel CLIP [29] embedding maps are unified for SAM-based [18] methods [28, 35] including ours. We follow the hyperparameter settings favorable to each respective paper.

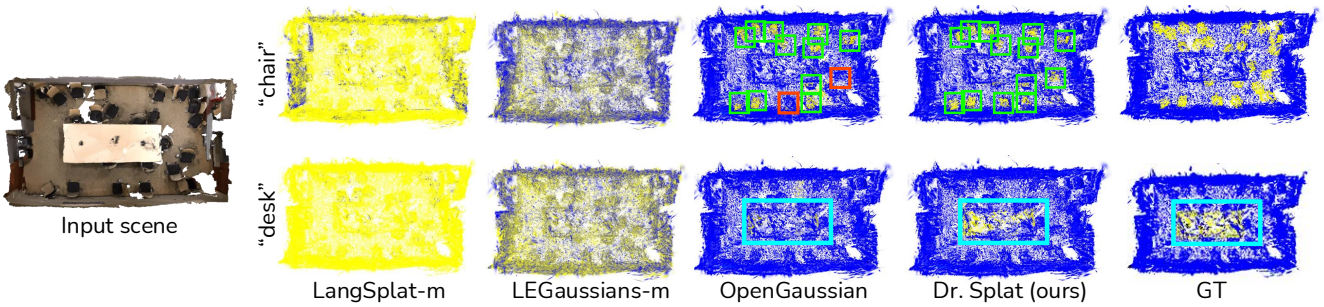


Figure 7. Qualitative results of 3D object localization. We visualize 3D localization activations (yellow) for "chair" and "desk" in the ScanNet dataset, comparing our method with others. It turns out that LangSplat-m and LEGaussians-m fail to localize objects accurately, while OpenGaussian struggles with object correspondence. Our model delivers precise and consistent localization across diverse queries.

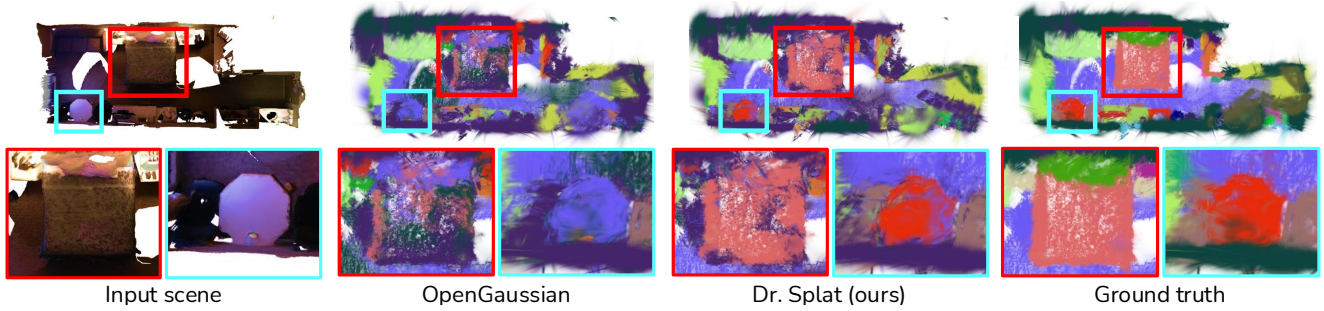


Figure 8. Visualization of open-vocabulary 3D semantic segmentation on the ScanNet dataset [4]. We visualize 3D Gaussian splat-based semantic segmentation using language features allocation of OpenGaussian [35] and Dr. Splat (ours) model on the same RGB-pretrained 3DGS. Note that, not specifically designed for segmentation, it achieves high performance as a result of language-based Gaussian updates.

4.1. 3D object selection

Settings. We first extract text features from an open-vocabulary text query using the CLIP model. Next, we compare text features to the 3D features embedded in each Gaussian using cosine similarity. By thresholding the similarity, we identify the 3D Gaussians that are relevant to the given text query. The selected 3D points are subsequently rendered into multi-view images using the 3DGS rasterization pipeline.

Results. We compare our model quantitatively with 3DGS-based language-embedded models as shown in Table 1. The results demonstrate that our method performs better object selection in most scenes, showing an improvement of over 0.5 in mIoU and more than 4.5 in mAcc compared to counterpart models. Notably, the rasterization-based method, LangSplat-m, often underperforms in most scenes.

Qualitative results are shown in Fig. 5. For LangSplat-m, the activations often shows random 3D Gaussians or fail to localize entirely (*e.g.*, see "coffee mug"), highlighting the limitations of rasterization-based methods and their unsuitability for 3D understanding, aligning the observation from Fig. 2. OpenGaussian frequently exhibits false activations with incorrect text-object pairs (*e.g.*, "apple" and "tea in a glass") and struggles to distinguish between nearby objects (*e.g.*, "waldo," "rubik's cube"). This artifacts can be attributed to use of spatial clustering and limited encoder capacity.

In contrast, our model leverages general image features thanks to the general PQ, maintaining feature distinctiveness regardless of scene complexity. Our feature registration considers the 3D geometry of the 3D Gaussians, which results in superior performance in 3D scene understanding tasks.

4.2. 3D object localization

Settings. Similar to the 3D object selection task, we calculate the cosine similarity between text query and 3D features embedded in each Gaussian. By thresholding the similarity, we identify the 3D Gaussians relevant to the given text query. To measure volume-aware localization evaluation, we propose a protocol to measure the IoU of 3D Gaussians that expands the traditional metric of point cloud-based approaches by incorporating volumetric information of 3D Gaussians.

Novel evaluation protocol for 3D localization in 3DGS. Unlike conventional evaluation protocol for the 3D localization task in point clouds, it is tricky to evaluate 3D localization performance in 3D Gaussians [16]. This is primarily

due to the un-deterministic structure of Gaussian distribution. To address this issue, we compute 3DGS pseudo-labels for evaluating the 3DGS localization in a volume-aware way. The details can be found in the supplementary material.

Given the ground truth, we measure IoU considering the spatial significance of each Gaussian and define a significant score d_i for each Gaussian θ_i with its scale $\mathbf{s}_i = [s_{ix}, s_{iy}, s_{iy}]$ and opacity α_i as $d_i = s_{ix}s_{iy}s_{iz}\alpha_i$, where $s_{ix}s_{iy}s_{iz}$ denotes a relative ellipsoid volume of a Gaussian θ_i. With the obtained significant scores $\mathbf{d} = [d_1, d_2, ..., d_N]$, we compute weighted IoU of 3D Gaussians to approximate volumes. The proposed metric is designed to assign a larger weight to the Gaussians with higher significant scores, when measuring IoU. Figure 6 shows that the impact of each Gaussian on the scene extremely varies depending on their significant scores, which demonstrates the necessity of the proposed IoU metric on 3D Gaussians that regards unequal contributions of each Gaussian.

Results. We report the 3D localization performance on the Scannet dataset in Table 2a. The 2D rasterization-based methods [28, 32] struggle to achieve precise activations for 3D localization. They inherently face challenges when applying for 3D tasks because they need to render 2D images for the scene interaction. Even with the 3D space search method, OpenGaussian [35], our model consistently demonstrates superior performance and achieves higher accuracy in localization. Figure 7 also shows that LangSplat-m and LEGaussians-m fail to properly localize the objects, and OpenGaussian misses queried objects in the scene.

4.3. 3D semantic segmentation

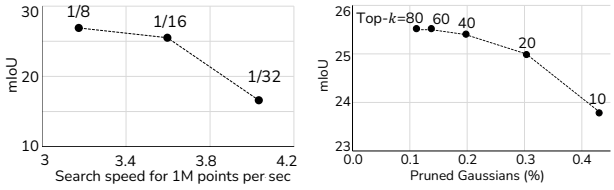
Settings. For a given set of open-vocabulary text labels, we perform segmentation by assigning each Gaussian a label having the highest activation among the known label set.

Results. The numerical comparison is presented in Table 2b. Although not explicitly designed for semantic segmentation, our model achieves notable performance in this task as a result of accurately updating each Gaussian with language features. Consistent with previous observations, rasterization-based 3DGS models exhibit lower segmentation performance. While OpenGaussian performs position-based clustering, our model demonstrates comparable performance, surpassing the baseline as the Top-k value increases. Our model also achieves better segmentation results, with a visual comparison of the segmented scene shown in Fig. 8.

4.4. Ablation study

We conduct an ablation study using the ScanNet dataset on different hyper-parameters of Dr. Splat to measure the contribution of each component.

Product Quantization. PQ introduces a trade-off between memory usage, computational efficiency, and accuracy. To better understand the balance between computational cost and localization quality, we conduct an ablation study by varying the number of sub-vectors. We evaluate performance at sub-vector sizes of 64, 128, and 256. Notably, these settings correspond to bit-size reductions of 1/32, 1/16, and 1/8 of the original CLIP feature, respectively. We measure the query distance computation time for one million data points, averaging results over 100 iterations for efficiency measure. Our findings reveal a favorable trade-off between quantization performance and accuracy (see Fig. 9-(b)) in the Pareto front with our PQ configurations. This achieves a balance that maximizes memory and computational efficiency while minimizing any loss in accuracy.



(a) Ablation on PQ parameters. (b) Ablation on Top-k.

Figure 9. Ablation study on (a) PQ and (b) Top-k Gaussians.

Top-k Gaussians. We examine the influence of the number of Gaussians assigned per ray. This parameter affects both memory requirements and computation, serving as a critical factor in overall performance. The ratio of pruned Gaussians and the mIoU results from different k are presented in Fig. 9a. We observe that increasing the aggregating number of Gaussians per ray improves localization performance; however, it results in higher memory consumption and the number of occupied Gaussians, indicating a clear trade-off.

5. Discussion and Conclusion

We present Dr. Splat, which is a novel approach for open-vocabulary 3D scene understanding by directly registering language embeddings to 3D Gaussians, eliminating the need for an intermediate rendering process. Compared to the previous 2D rendering-based methods [28, 32], which have limited search domain and capacity, our method directly searches 3D space while preserving the fidelity of language embeddings. This operation is further accelerated by the integration of Product Quantization (PQ)

Experimental results validate Dr. Splat 's superior performance across various 3D scene understanding tasks, including open-vocabulary 3D object selection, 3D object localization, and 3D semantic segmentation. These findings highlight Dr. Splat's ability to transform 3D scene understanding by achieving a balance between highly representative quality and computational efficiency. This breakthrough paves the way for advanced applications in robotics, autonomous navigation, and augmented reality.

Acknowledgment

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) (No.RS-2025-25443318; Physically-grounded Intelligence: A Dual Competency Approach to Embodied AGI through Constructing and Reasoning in the Real World; No.RS-2019-II191906, Artificial Intelligence Graduate School Program(POSTECH); No. RS-2024-00397663, Real-time XR Interface Technology Development for Environmental Adaptation, 25%), and by Electronics and Telecommunications Research Institute (ETRI) [25ZD1160, Development of ICT Convergence Technology for Daegu-Gyeongbuk Regional Industry] grant funded by the Korea government(MSIT)

References

[1] 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 IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 2

[2] Jaesung Choe, Chunghyun Park, Francois Rameau, Jaesik Park, and In So Kweon. Pointmixer: Mlp-mixer for point cloud understanding. In *European Conference on Computer Vision*, pages 620–640. Springer, 2022. 1

[3] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3075–3084, 2019. 1

[4] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, pages 5828–5839, 2017. 6, 7

[5] 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. 2

[6] Francis Engelmann, Fabian Manhardt, Michael Niemeyer, Keisuke Tateno, Marc Pollefeys, and Federico Tombari. OpenNerf: Open Set 3D Neural Scene Segmentation with Pixel-Wise Features and Rendered Novel Views. In *ICLR*, 2024. 2

[7] Michael R. Garey and David S. Johnson. Computers and intractability. a guide to the theory of np-completeness. *W. H. Freeman and company*, 174, 1979. 3

[8] Antoine Guédon, Tom Monnier, Pascal Monasse, and Vincent Lepetit. Macarons: Mapping and coverage anticipation with rgb online self-supervision. In *CVPR*, 2023. 3

[9] Jun Guo, Xiaojian Ma, Yue Fan, Huaping Liu, and Qing Li. Semantic gaussians: Open-vocabulary scene understanding with 3d gaussian splatting. *arXiv preprint arXiv:2403.15624*, 2024. 2

[10] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019. 5

[11] Zhening Huang, Xiaoyang Wu, Xi Chen, Hengshuang Zhao, Lei Zhu, and Joan Lasenby. Openins3d: Snap and lookup for 3d open-vocabulary instance segmentation. In *European Conference on Computer Vision*, pages 169–185. Springer, 2025. 1, 3

[12] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Alaa Maalouf, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, et al. Conceptfusion: Open-set multimodal 3d mapping. *arXiv preprint arXiv:2302.07241*, 2023. 2

[13] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2010. 5

[14] Yuzhou Ji, He Zhu, Junshu Tang, Wuyi Liu, Zhizhong Zhang, Yuan Xie, and Xin Tan. Fastlgs: Speeding up language embedded gaussians with feature grid mapping. *arXiv preprint arXiv:2406.01916*, 2024. 2, 3

[15] Li Jiang, Shaoshuai Shi, and Bernt Schiele. Open-vocabulary 3d semantic segmentation with foundation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21284–21294, 2024. 1, 2

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM TOG*, 2023. 1, 2, 3, 7

[17] 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, 5, 6

[18] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloé Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross B. Girshick. Segment anything. In *ICCV*, 2023. 4, 6

[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. 2

[20] Hyunjee Lee, Youngsik Yun, Jeongmin Bae, Seoha Kim, and Youngjung Uh. Rethinking open-vocabulary segmentation of radiance fields in 3d space, 2024. 2

[21] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and René Ranftl. Language-driven semantic segmentation. *arXiv preprint arXiv:2201.03546*, 2022. 2

[22] 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. 2

[23] Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su. Partslip: Low-shot part segmentation for 3d point clouds via pretrained image-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21736–21746, 2023. 2

[24] 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. 2

[25] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 815–824, 2023. 1, 2, 4

[26] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 1

[27] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in neural information processing systems*, 35:23192–23204, 2022. 1

[28] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20051–20060, 2024. 1, 2, 3, 4, 5, 6, 8

[29] 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, 6

[30] Adam Rashid, Satvik Sharma, Chung Min Kim, Justin Kerr, Lawrence Yunliang Chen, Angjoo Kanazawa, and Ken Goldberg. Language embedded radiance fields for zero-shot task-oriented grasping. In *7th Annual Conference on Robot Learning*, 2023. 2

[31] Damien Robert, Hugo Raguet, and Loic Landrieu. Efficient 3d semantic segmentation with superpoint transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17195–17204, 2023. 1

[32] 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. 1, 2, 3, 4, 6, 8

[33] 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. 1

[34] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4840–4851, 2024. 1

[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. 1, 2, 3, 4, 5, 6, 7, 8

[36] Jihan Yang, Runyu Ding, Weipeng Deng, Zhe Wang, and Xiaojuan Qi. Regionplc: Regional point-language contrastive learning for open-world 3d scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19823–19832, 2024. 1, 2

[37] 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. 2

[38] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021. 1

[39] Yuhang Zheng, Xiangyu Chen, Yupeng Zheng, Songen Gu, Runyi Yang, Bu Jin, Pengfei Li, Chengliang Zhong, Zengmao Wang, Lina Liu, Chao Yang, Dawei Wang, Zhen Chen, Xiaoxiao Long, and Meiqing Wang. Gaussiangrasper: 3d language gaussian splatting for open-vocabulary robotic grasping. *IEEE Robotics and Automation Letters*, 2024. 1, 2, 3

[40] 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.

[41] Xingxing Zuo, Pouya Samangouei, Yunwen Zhou, Yan Di, and Mingyang Li. Fmgs: Foundation model embedded 3d gaussian splatting for holistic 3d scene understanding. *IJCV*, 2024. 1, 2, 3

[42] Matthias Zwicker, Hanspeter Pfister, Jeroen van Baar, and Markus H. Gross. EWA volume splatting. In *Visualization Conference*, 2001. 2