

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

— Supplementary Material —

Kim Jun-Seong¹ GeonU Kim² Kim Yu-Ji² Yu-Chiang Frank Wang³
Jaesung Choe^{3*} Tae-Hyun Oh^{1,2,4*}

¹Department of Electrical Engineering and ²Graduate School of AI, POSTECH

³NVIDIA ⁴School of Computing, KAIST

{junseong.kim, gukim, ugkim}@postech.ac.kr

{frankwang, jchoe}@nvidia.com taehyun@kaist.ac.kr

Contents

A Implementation Details	1
B Experiment Setup	2
C Evaluation Protocols	3
D Analysis on Product Quantization	5
D.1. Search-time Experiments	5
D.2. Scalability of the PQ codebook.	5
E Additional Results	6
E.1. Additional results on presented 3D tasks . . .	6
E.2. Experiments on the ScanNet200 dataset . . .	6
E.3. Experiments on the city-scale dataset	6
F. Broader Applications and Limitations	8

Supplementary Material

In this supplementary material, we provide additional details omitted from the manuscript. Sec. A covers implementation and evaluated 3D tasks. Sec. B outlines the experimental setup, and Sec. C explains our Gaussian-friendly evaluation protocol. Sec. D presents additional comparison experiments on pq codebook, while Sec. E includes qualitative results, annotation analyses, and city-scale dataset evaluations. Sec. F addresses limitations and future directions. We also provide a supplementary video that highlights city-scale experiments.

A. Implementation Details

Overall, our method consists of (1) a pre-processing stage that constructs the codebooks in Product Quantization,

and pre-training of 3DGS, (2) a training stage that aggregates multiview CLIP embeddings into unified Gaussian-registered embeddings, and (3) Inference stage that directly referring to language-embedded 3D Gaussians for the downstream task.

Pre-Training stage. In the pre-processing stage, we need to extract per-patch CLIP embeddings to build PQ codebooks. It consists of a patch extraction step, and a CLIP embedding extraction step. To obtain patches, we utilize the LVIS dataset, a large-scale dataset having ground truth image segmentations. From the segmentation mask given in the LVIS dataset, we identify object regions and crop them into individual image patches. Each cropped patch is then processed and encoded using the OpenCLIP ViT-B/16 model. Based on these predictions, we continue to build PQ codebooks. We utilize FAISS [3] open-source library for our Product Quantization implementation. We use 128 sub-vectors per embedding, with each subvector assigned to one of 256 centroids, yielding an 8-bit index per subvector. This process is illustrated in Fig. S5.

3D Gaussian parameters Θ [7] are also optimized during the pre-processing stage. We typically care about this initialization of the 3D Gaussians, which can potentially impact the performance of the 3D scene understanding tasks. So, we follow the original 3D Gaussian Splatting method and utilize the optimized 3D Gaussians as our initial parameters. In other words, the pre-training is conducted using the default hyperparameters from 3DGS [7] framework, running 30,000 iterations. Also, we consistently apply this paradigm across different methods for fair comparison. Especially, for LeGaussian [20] that employs mutual training, we disabled 3D Gaussian updates during feature assignment in our experiments.

Training stage. Based on the PQ and the initial 3D Gaussian parameters Θ , we begin the training stage. All competing

*Corresponding Authors

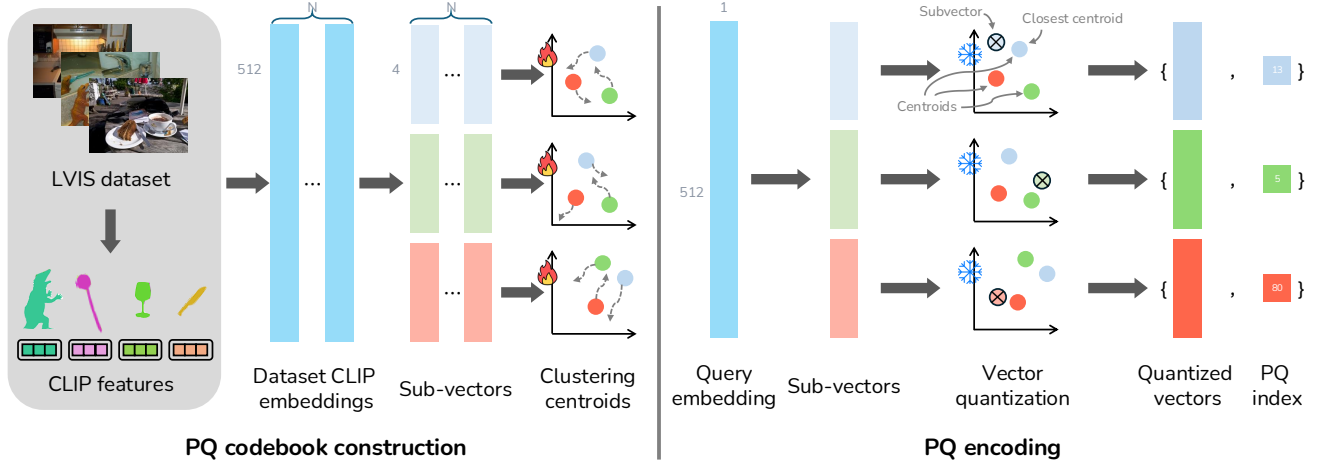


Figure S1. We illustrate the process of construction and encoding process of Product Quantization we used in Dr. Splat. (left) We first construct by update subvector centroids using CLIP features extracted from large-scale object images. (right) After constructing PQ codebook, centroids for each sub-vectors are kept frozen. For query feature, we divide into sub-vectors and are encoded into centroid indices by finding nearest neighbor.

models and our proposed model are trained and evaluated on a single NVIDIA RTX A6000 GPU to ensure fair performance comparison. The training stage consists of three main steps: extracting pixel-wise CLIP embeddings from training images, the feature aggregation stage, and lastly feature registration stage.

Given multi-view images, we extract dense CLIP features assigned to each pixel. To obtain per-pixel CLIP features, we adopt the feature extraction scheme by LangSplat [16], which utilizes SAM [12]. To collect per-patch embeddings, we followed the OpenGaussian framework and used a single-level mask, while LangSplat utilized multi-level masks.

Once these CLIP features are extracted for all images, we proceed to the feature registration step. In the feature registration step, we iteratively measure the contribution (weights) of pre-trained Gaussians for each ray assigned to the pixels in the training images, and update the 3D Gaussian embeddings. These weights are determined according to the volume rendering equation, which defines their influence during the color rasterization process (see Sec. 3.2 of the manuscript). After the registration process, we normalize the embeddings by dividing embeddings with L2 norms.

Lastly, we register aggregated features to 3D Gaussians. For memory efficiency, we quantize the aggregated Gaussians using the pre-trained PQ codebooks to encode features to indices, a set of 128-channel 8-bit integer indices (Fig. S5). While registration, Gaussians that were never selected in the top-k process are pruned to reduce noise and memory consumption. At the end of the training, we retain a set of assigned Gaussians with 128 8-bit integer indices.

Inference stage. Finally, in the inference stage, the PQ-assigned Gaussians from the previous steps are used. By

recalling the PQ index list assigned to each 3D Gaussian, cosine similarity is computed between the embeddings of a given text query, extracted using the same CLIP encoder, and each 3D Gaussian. Detailed steps are provided in Sec. 3.3 of the manuscript. As the subvector norms do not sum to 1, normalization by the sum of the subvector L2 norms is applied. We can apply this by using the `search` function in the Faiss library. The resulting similarity scores are then used to perform various 3D tasks, as evaluated in the study. The following sections explain how the computed activation values are applied in each task.

B. Experiment Setup

We conduct experiments on three different tasks: 3D object selection task, open-vocabulary 3D object localization task, and open-vocabulary 3D semantic segmentation task. These tasks are closely related to the 3D search as described in Fig. 1 of the manuscript as well as the 3D scene understanding tasks [22].

3D object selection. To evaluate the model’s 3D awareness capability, we evaluate a 3D object selection task. We first extract text features from an open-vocabulary text query using the CLIP text encoder [17]. Next, we compare these text features to the 3D Gaussian embeddings by computing the cosine similarity score. By thresholding the similarity, we identify the 3D Gaussians that are relevant to the given text query. The threshold value for each method is determined through a grid search to identify the optimal performance.

We use the LeRF-OVS dataset [8] with annotations by LangSplat [16]. As the LeRF-OVS dataset lacks 3D ground truth, we follow the 2D segmentation-based evaluation method proposed by OpenGaussian [22]. This approach

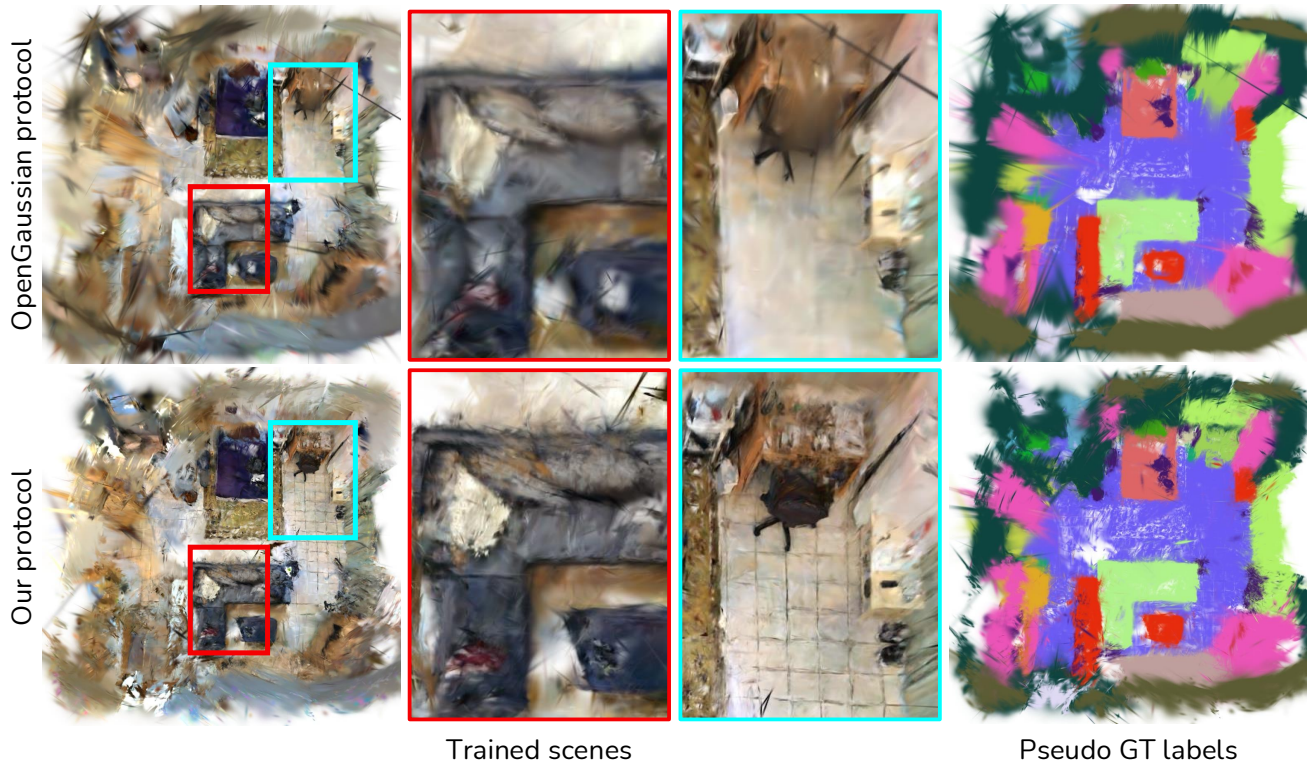


Figure S2. We compare the quality of the scenes and pseudo ground truth labels obtained from different evaluation protocols. (top) Trained scenes following OpenGaussian evaluation protocol, which fix the positions and the number of the initial points during training. (bottom) Trained scenes following our evaluation protocol, which dose not require any constraint during training.

evaluates 3D understanding by measuring multi-view 2D segmentation accuracy between the rendered occupancy mask from the selected 3D Gaussians and the GT object masks. Ground truth segmentation masks are manually annotated corresponding to text queries as described in [16]. We evaluate the IoU and localization accuracy for the metric.

Open-vocabulary 3D object localization. Given an open-vocabulary text query, we use the CLIP text encoder [17] to extract a text feature of the given text query. Then, we compute the cosine similarity score between the query text feature and the Gaussian-registered embeddings. Finally, we select highly relevant 3D Gaussians by thresholding the obtained cosine similarities. We set the threshold of each method individually by searching the thresholds that show the best mIoU on the scenes used for evaluation.

Open-vocabulary 3D semantic segmentation. We further evaluate our method using the open-vocabulary 3D semantic segmentation task. For a given set of open-vocabulary text queries representing categories, we use the CLIP text encoder to extract a language embedding for each query. We then compute the cosine similarity scores between the 3D Gaussian embeddings and the language features from

the given text queries. Using the obtained cosine similarity scores, we assign each 3D Gaussian to the category with the highest the cosine similarity score.

C. Evaluation Protocols

Limitations of existing evaluation protocols. Compared to the previous works, such as LERF [8], LEGaussian [20], and LangSplat [16], our method challenges to leverage the 3D Gaussian representation into the 3D scene understanding tasks. Similar to ours, OpenGaussian [22] is a concurrent work that aims at the open-vocabulary 3D semantic segmentation task as well. However, unlike OpenGaussian, we introduce a new evaluation criterion specialized for the 3D Gaussians, instead of using point cloud-specific evaluations.

OpenGaussian [22] computes evaluation metrics directly from 3D Gaussians, using ScanNet [2] ground truth point clouds with semantic labels. It aligns Gaussian centers μ with dataset points $[x, y, z]$ and keeps both μ and the number of Gaussians N fixed during parameter optimization. This differs from vanilla 3D Gaussian Splatting [7]. As shown in Fig. S2, their approach introduces significant quality issues, influenced by the evaluation metric. However, the

reason behind this optimization trick is related to evaluation.

The evaluation by OpenGaussian involves predicting labels for each Gaussian and measuring their alignment with the ground truth point cloud using Intersection over Union (IoU). To compute IoU, the overlap (intersection) and total extent (union) of the points are calculated between the 3D Gaussians' center locations $\{\mu\}$ and the ground truth point clouds at fixed positions. As we discussed, since OpenGaussian does not update the locations of the 3D Gaussians, which is identical to the locations of the 3D ground truth points, they simply count the overlap and union without considering the volumetric properties of the 3D Gaussians.

We claim that such an evaluation protocol has two dominant issues. First, by pre-defining the number of Gaussians as well as the center locations of the 3D Gaussians, the optimized 3D Gaussians produce degraded rendering quality as shown in Fig. S2, which is not a practical solution. Second, the aforementioned IoU is calculated only with the number of 3D Gaussians, which does not consider the significance of each Gaussian having different shapes and densities.

Our Gaussian-friendly evaluation protocol. To address these limitations, we propose a novel evaluation protocol to compute IoU from 3D Gaussians. Our evaluation protocol follows the original 3D Gaussian Splatting's optimization scheme [7] by updating the location of the 3D Gaussians as well as the number of 3D Gaussians. After we obtain the optimized Gaussians Θ , these parameters are used to train language-embedded Gaussians. Then, the following question is how we assign the ground truth semantic labels for each Gaussian from the existing per-point semantic annotations provided by the ScanNet dataset [2].

Starting from the given Q numbers of point cloud $\mathcal{P} = \{\mathbf{p}_k\}_{k=1}^Q$ and a set of semantic labels $\mathcal{S} = \{\mathbf{s}\}$, we compose a paired set of points and their labels as $\{\mathbf{p}_k, \mathbf{s}^{\mathbf{p}_k}\}_{k=1}^Q$, which is provided by the official datasets. We measure the Mahalanobis distances between the language-embedded 3D Gaussian parameters $\Phi = \{\theta_i, \tilde{\mathbf{f}}_i\}_{i=1}^N = \{\mu_i, S_i, R_i, \alpha_i, \mathbf{c}_i, \tilde{\mathbf{f}}_i\}_{i=1}^N$ (Sec. 2 of the manuscript) and ground truth point clouds. Note that the Mahalanobis distances is already used in the 3DGS [7] when computing effective alpha values, as stated in the Eq. 1 of the manuscript. We maintain to use this equation to calculate the Mahalanobis distance $\mathbf{d}^{\text{mahal}}(\cdot)$ between volumetric 3D Gaussian θ and 3D point \mathbf{p} as:

$$\mathbf{d}^{\text{mahal}}(\mathbf{p}, \theta) = (\mathbf{p} - \mu)^\top \Sigma^{-1} (\mathbf{p} - \mu). \quad (1)$$

Using the Mahalanobis distance, we determine the semantic label of each Gaussian as below:

$$\mathbf{s}^{\theta_i} = \arg \max_{\mathbf{s} \in \mathcal{S}} \left(\sum_{\mathbf{p}_k \in \mathcal{P}} \mathbb{1}\{\mathbf{s}^{\mathbf{p}_k} = \mathbf{s}\} \cdot \mathbf{d}^{\text{mahal}}(\mathbf{p}_k, \theta_i) \right), \quad (2)$$

where \mathbf{s}^{θ_i} is the semantic label of the i -th 3D Gaussian θ_i , $\mathbb{1}\{\mathbf{s}^{\mathbf{p}_k} = \mathbf{s}\}$ is an indicator function returning 1 only when

k -th point label is identical to a semantic label $\mathbf{s} \in \mathcal{S}$. In shorts, this equation determines the semantic label of each 3D Gaussian from the specific semantic label \mathbf{s} that has the highest sum of the Mahalanobis distances from the ground truth point to each 3D Gaussian.

The proposed assignment process enables generally applicable evaluation of 3D Gaussians without any constraints. Fig. S2 shows the quality degradation of the trained scene following the OpenGaussian evaluation protocol, which fixes the position and the number of initial points during training. On the other hand, our generalizable evaluation protocol does not impose any constraints during the training of Gaussians, and it also enables high-quality scene reconstruction, effectively capturing detailed areas.

With the obtained N number of pseudo GT 3D Gaussians, we measure IoU by considering the volumetric significance of each Gaussian. We define the significant score d_i for each Gaussian θ_i with its scale $\mathbf{s}_i = [s_{ix}, s_{iy}, s_{iz}]^\top$ 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 $\mathbf{d} = [d_1, d_2, \dots, d_N]^\top$, we calculate IoU of i -th 3D Gaussians for the label as:

$$\begin{aligned} \text{Intersection}_i &= \mathbf{d} \cdot (\mathbf{l}_i^{\text{pred}} \odot \mathbf{l}_i^{\text{gt}}), \\ \text{Union}_i &= \mathbf{d} \cdot (\mathbf{l}_i^{\text{pred}} + \mathbf{l}_i^{\text{gt}} - (\mathbf{l}_i^{\text{pred}} \odot \mathbf{l}_i^{\text{gt}})), \\ \text{IoU}_i &= \text{Intersection}_i / \text{Union}_i, \end{aligned} \quad (3)$$

where $\mathbf{l}_i^{\text{pred}} \in \mathbb{R}^N$ and $\mathbf{l}_i^{\text{gt}} \in \mathbb{R}^N$ are binary vectors indicating whether the predicted/GT label of each Gaussian is the n -th label, \mathbf{s}^θ in Eq. (2). The proposed metric is designed to assign a larger weight to the Gaussians with higher significant scores when measuring IoU, and the significant score endows our metric with volume-awareness.

Volume awareness of the proposed metric. To validate that the proposed metric can effectively approximate the volumetric IoU of the 3D scene, we compare our metric with another volume-aware IoU measurement based on voxel representation. Before measuring IoU with voxels, we train 3D Gaussians and generate labeled pseudo-GT 3D Gaussians with Eq. (2). Then we first sample voxels in the scene, and allocate a GT label to each voxel with the labeled 3D Gaussians. We obtain the most likely label of each voxel by defining the label score. The label score l_{jn}^{voxel} is computed with the opacity α_i and the density $\mathcal{N}(\mathbf{v}_j | \mu_i, \Sigma_i)$ of each Gaussian at the position of a voxel \mathbf{v}_j as:

$$l_{jn}^{\text{voxel}} = \sum_{\theta_i \in \Theta} \alpha_i \cdot \mathbb{1}\{\mathbf{s}^{\theta_i} = \mathbf{s}\} \cdot \mathcal{N}(\mathbf{v}_j | \mu_i, \Sigma_i), \quad (4)$$

where $\mathbb{1}\{\mathbf{s}^{\theta_i} = \mathbf{s}\}$ is an indicator function determining whether a Gaussian θ_i is assigned to the n -th label and $\det(\Sigma_i)$ is the determinant of Σ_i . With the obtained score, we first filter out empty voxels by thresholding with: $p_j =$

	OpenGaussian evaluation		Our evaluation	
	OpenGaussian	Ours	OpenGaussian	Ours
IoU > 0.15	52.7	54.3	57.8	52.6
IoU > 0.30	36.4	39.4	38.0	40.3
IoU > 0.45	14.7	15.5	18.3	25.6
3D mIoU	23.1	25.0	25.2	25.4

Table S1. We compare different metrics for measuring IoU, proposed by OpenGaussian [22] and our work.

$\sum_{n=1}^L l_{jn}^{\text{voxel}}$, where L is the total number of the labels, which can be interpreted as a density of each voxel \mathbf{v}_j . Then we assign a label with the highest score, as the GT label of each voxel. We can also generate predicted labels of voxels using the predicted labels of Gaussians in the same manner, and can evaluate IoU by comparing the GT and predicted labels of the voxels one-to-one.

Volume awareness is inherent in this voxel-based IoU evaluation as the voxels explicitly represent the volume of the scene. We show the volume-awareness of our evaluation metric by showing a correlation between our metric and voxel-based metric in Fig. S3. As can be seen, our metric obtains a high correlation with the voxel-based IoU evaluation metric by considering the significant score when calculating IoU. This result shows the necessity of the significant score, which endows our metric with volume awareness.

Although the voxel-based IoU evaluation effectively measures volume-aware IoU of the scene, the computational cost to assign labels is too expensive. Each time new labels of Gaussians are predicted, the process of assigning them to the voxels is required for evaluation. Different from voxel-based IoU evaluation, our IoU evaluation protocol has a low computational cost, since there is no repeated assignment process after we once generate the labeled pseudo-GT Gaussians. In other words, our proposed IoU evaluation protocol is a fast and volume-aware evaluation for measuring the IoU of scenes represented by 3D Gaussians.

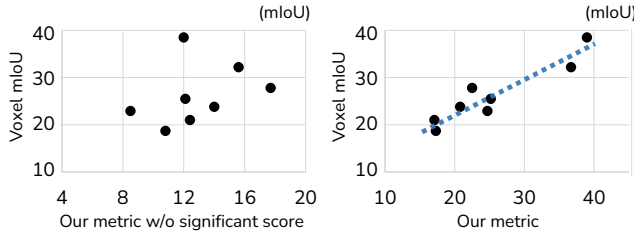


Figure S3. Scatter plot of mIoUs with different mIoU evaluation protocols, measured from eight scenes of the ScanNet [2] dataset. (left) Low correlation between voxel-based metric and our metric without significant score, *i.e.*, same score d_i for all Gaussians. (right) High correlation between voxel-based metric and our metric.

D. Analysis on Product Quantization

D.1. Search-time Experiments

In addition to its memory efficiency, Product Quantization significantly enhances search speed. Product quantization can approximate distances between vectors using quantized sub-vectors. By precomputing and storing distances between subvector centroids in a Look-Up Table (LUT), distance calculations between query and database vectors during the search phase are reduced to simple indexing operations. The precomputation shifts the complexity of vector distance calculations from $O(ND)$ for a D dimensional vector to $O(N)$ per subvector.

Use of LUT can be described as follows. For trained PQ centroids c_{lj} , for $l = 1, 2, \dots, L$, and $j = 1, \dots, 2^k$, where L is number of sub-vectors, and k refers the number of bits used for indexing each centroids. LUT is stored as follows:

$$\text{LUT}_l[i, j] = \|c_{li} - c_{lj}\|_2^2, \text{ where } i, j \in \{1, 2, \dots, 2^k\}. \quad (5)$$

Then for vectors, $\mathbf{v}_1 = [\mathbf{v}_{11}, \dots, \mathbf{v}_{1L}]$, $\mathbf{v}_2 = [\mathbf{v}_{21}, \dots, \mathbf{v}_{2L}]$ mapped to indices $j_1 = [j_{11}, j_{12}, \dots, j_{1L}]$, and $j_2 = [j_{21}, j_{22}, \dots, j_{2L}]$, distance is computed as summation of each retrieved LUT values following each PQ indices:

$$d(v_1, v_2) = \sum_{l=1}^L \text{LUT}_l(j_{1l}, j_{2l}). \quad (6)$$

We can also compute the cosine similarity of the vectors, by computing inner products rather than distances, following normalization by the sum of each norm of each sub-vector. Despite the quantization errors, previous literature [5] shows that these errors remain within certain quantization bounds, preserving the correlation between the approximated and actual distances.

The scalability and speed of the proposed approach make it particularly suitable for handling complex 3D data. We compared search speed between computing cosine similarity of CLIP features and distance computation in product quantization (see Fig. S4). Under identical hardware conditions, the proposed LUT-based approach demonstrated substantial speed improvements compared to cosine similarity computation between CLIP features: with a subvector size of 128, 64, 32 search performance improved by approximately $2\times$, $6.6\times$, $14.1\times$ respectively. These improvements underscore the computational advantages of the proposed method.

Considering that rendering-based methods require significantly greater computation compared to 3D data processing, Dr. Splat’s approach demonstrates its superior efficiency in search efficiency, and establishes itself as a practical and scalable solution for 3D data search and processing at scale.

D.2. Scalability of the PQ codebook.

Our PQ codebook provides a scalable, scene-agnostic compression method. Unlike per-scene codebook optimization,

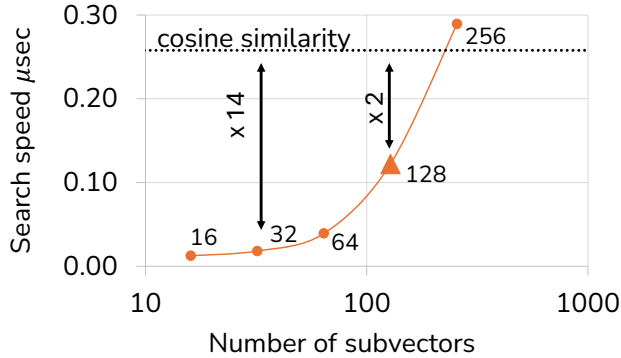


Figure S4. We compare inference speed between product quantization LUT based method and ordinary cosine similarity calculation. We calculate average inference time spent over one million feature points. We report mean values over 100 repeated experiments

PQ requires no extra training to new scenes, ensuring immediate applicability and versatility for compression tasks. We already analyzed the trade-off between compression size and 3D task performance in Sec. 4.4, and Fig. 9-(a) in main. To further evaluate information loss with traditional codebook, we compare reconstruction accuracy during compression in Table S2. We train LSH (random projection) [1] and LangSplat’s [16] AutoEncoder (AE) on CLIP features from the LVIS dataset. For evaluation, we test on the LeRF dataset by calculating cosine similarity between the original CLIP features extracted from the test dataset and the reconstructed features using each codebook. For same compression rate, ours shows superior reconstruction accuracy, which demonstrates the effectiveness and scalability of PQ codebook.

Codebook	1/32	1/16	1/8
LSH	79.71 (-11.15)	88.35 (-7.70)	93.69 (-5.33)
AE	82.58 (-8.28)	84.03 (-12.03)	85.18 (-13.84)
PQ (ours)	90.86	96.06	99.02

Table S2. **Reconstruction accuracy on LeRF dataset.** {1/32, 1/16, 1/8} are the compression ratios. AE denotes AutoEncoder.

E. Additional Results

In this section, we present additional results that are not shown in the manuscript due to space constraints.

E.1. Additional results on presented 3D tasks

We first show more experimental results for the 3D object selection task in Fig. S6, and the 3D localization task in Fig. S7 which are not included in the manuscript due to the space limit. Consistent with our earlier observations, the LangSplat model struggles to learn accurate 3D features. While it occasionally follows feature patterns, it frequently produces significant noise, making it unsuitable for real-world applications such as localization, object grabbing, or

	3D mIoU	200 classes		
		IoU > 0.15	IoU > 0.3	IoU > 0.45
LangSplat-m [16]	3.9	7.6	3.5	0.8
LEGaussians-m [20]	4.0	7.4	3.8	1.4
OpenGaussian [22]	14.7	34.2	18.9	11.0
Ours (Top-20)	14.6	36.3	18.6	9.4
Ours (Top-40)	14.9	36.0	19.3	14.0

Table S3. We compare evaluate our method with previous methods on the ScanNet-200 dataset.

3D image editing. Additionally, we observe persistent spatial bias in the OpenGaussian method, as previously noted, see red cup, plate, or wavy noodles, and bed cases in Fig. S7, it fails to select relevant regions in others. In contrast, our proposed method, which allows direct search and inference in 3D space, consistently identifies favorable localization performance. This demonstrates the robustness and practicality of our approach compared to competing methods.

In the Sec. C, we demonstrated that our metric provides superior volumetric alignment compared to existing approaches. To further validate the superiority of our model, we also evaluated its performance using the metric proposed by OpenGaussian. We confirm that our method outperforms even using other evaluation protocols as shown in Table S1

E.2. Experiments on the ScanNet200 dataset

The proposed model and its counterparts are designed to operate effectively in open-vocabulary settings. To evaluate performance under more comprehensive open-vocabulary cases, we conducted additional experiments using the ScanNet-200 annotation [19], which extends the ScanNet limited-label of 20 to 200 semantic categories, including tail categories such as armchair and windowsill. These rare classes provide a closer approximation to real-world scenarios and enable a robust assessment of the models’ generalization capabilities. For consistency, experiments are conducted using the same scenes as previous benchmarks, following ground truth annotations as described in Sec. C.

The results, summarized in Table S3, demonstrate that the proposed model consistently outperforms its counterparts, which highlights superior generalization across diverse object spaces. The results validate the proposed model’s ability to excel across both constrained and diverse object spaces, emphasizing its potential for practical application in complex real-world scenarios.

E.3. Experiments on the city-scale dataset

As suggested in the previous work [9–11], the proposed method is further evaluated in a large-scale scenario using the Waymo San Francisco Mission Bay dataset [21], which features expansive spatial contexts. For each scene, the dataset comprises approximately 12,000 images captured by 12 cameras, providing a challenging and diverse testing

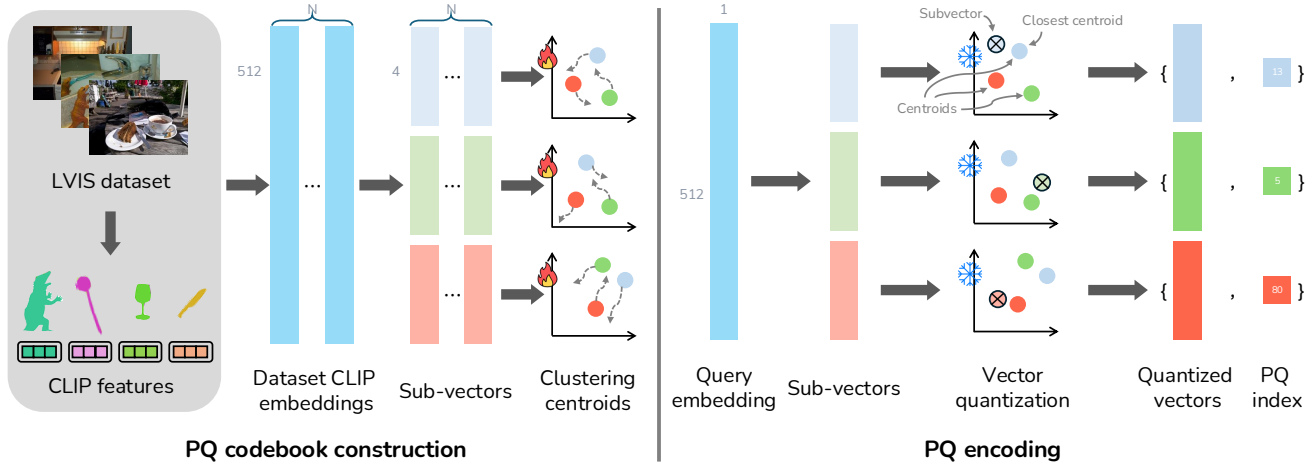


Figure S5. We illustrate the process of construction and encoding process of Product Quantization we used in Dr. Splat. (left) We first construct by update subvector centroids using CLIP features extracted from large-scale object images. (right) After constructing PQ codebook, centroids for each sub-vectors are kept frozen. For query feature, we divide into sub-vectors and are encoded into centroid indices by finding nearest neighbor.

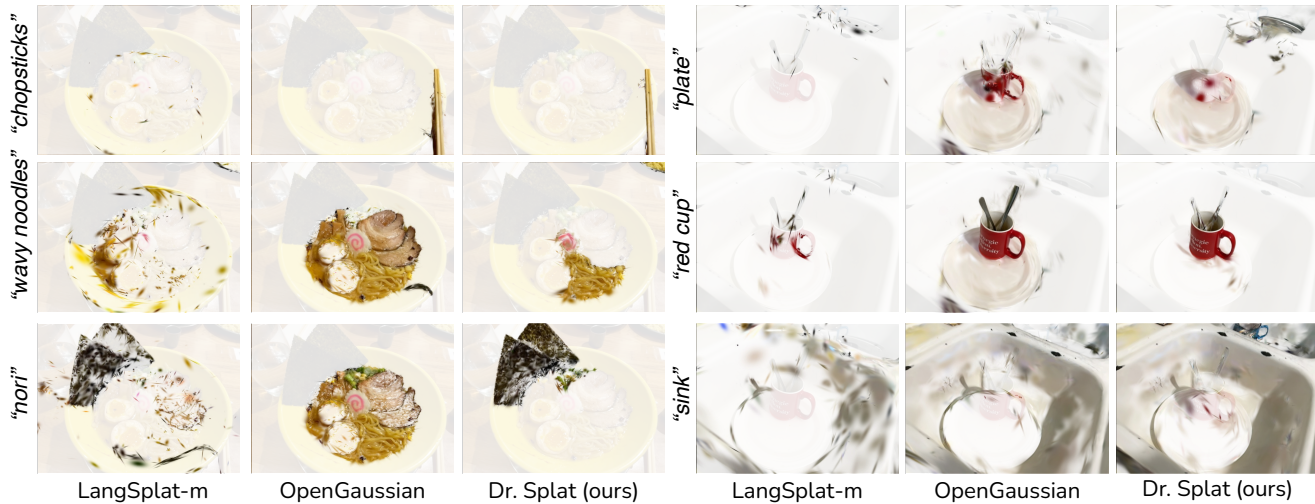


Figure S6. We compare 3D object selection task in LeRF dataset with Langsplat [16], and OpenGaussian [22]. We visualize selected Gaussians with high similarity to query text. Langsplat shows noisy, 3D uncorrelated activations, and Opengaussian often show false positive activations, while our method show accurate localization showing superiority on generalizability.

environment for 3D localization tasks. We select 3 blocks of the scene for large-scale scene tests.

We conducted comparisons against the LangSplat-m model for the 3D text-query localization task as shown in Fig. S8. Our evaluation focused on qualitatively assessing how well each model performs in localizing queries within the 3D space. Our method consistently succeeds in localizing diverse text queries, demonstrating robust and accurate performance across various contexts. In contrast, LangSplat-m struggles to make precise predictions, particularly with its 3D Gaussian representations failing to align with the expected ground-truth values. These findings are consistent with our earlier observations regarding the limitations of

LangSplat-m’s approach.

As shown in Fig. S9, we can see that the results reflect not only objects, but also attributes like color to some extent. Additional visualizations of the results can be found in Fig. S10 and the supplementary video, which provides a more comprehensive view of the qualitative differences between the methods. We strongly encourage readers to refer to these supplementary materials for further insights.

The differences between the methods become even more pronounced when considering search speed in large-scale scenarios. For example, the Waymo dataset contains over 2.9M Gaussians, with individual images requiring nearly 1M computations per image for over 100 images. The computa-

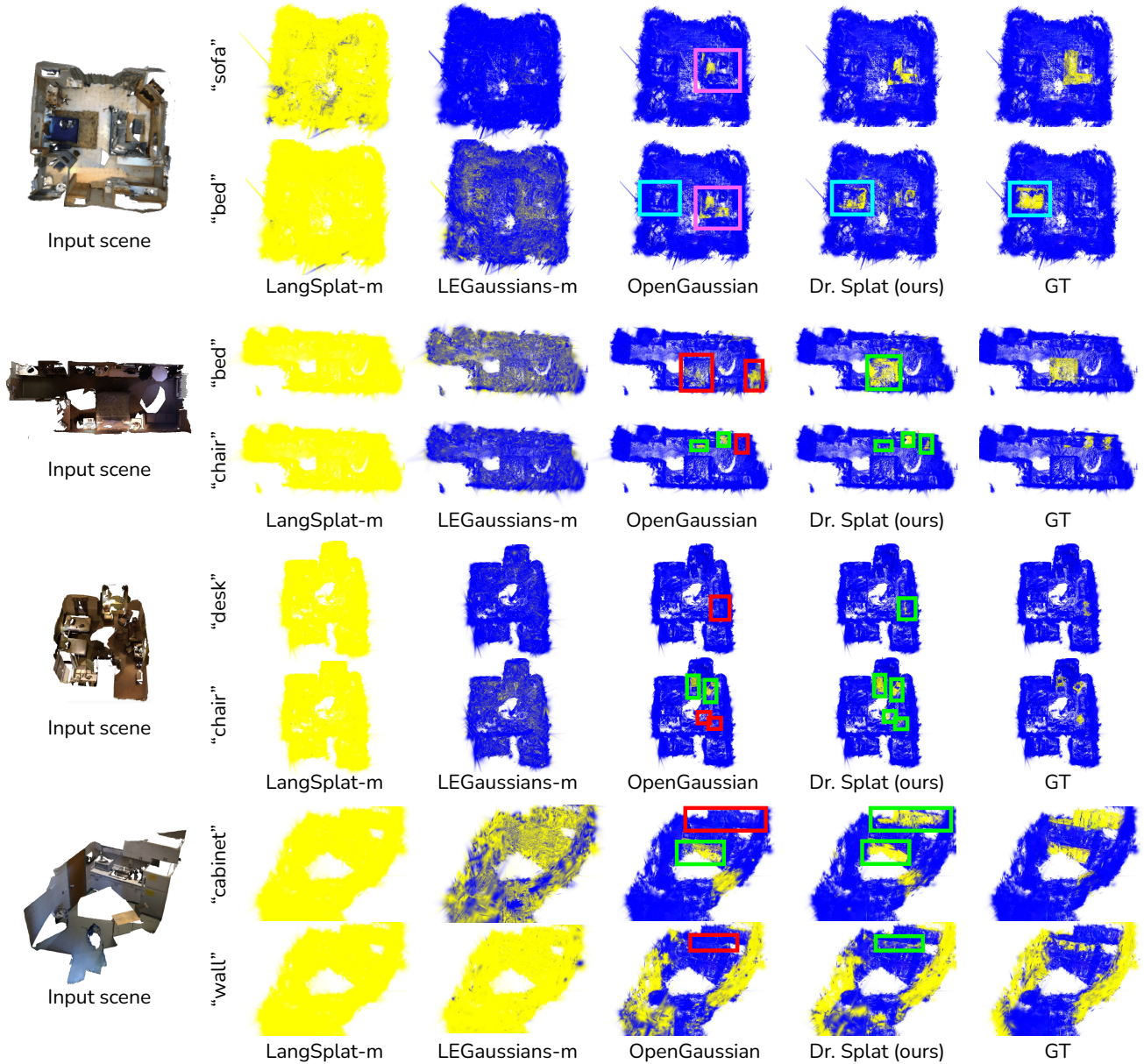


Figure S7. We compare 3D object localization results between competing methods [16, 20, 22] with Dr. Splat. 3DGS with similarity above the threshold (0.562) are shown in yellow, while those below the threshold are displayed in blue. Greenbox indicates successful localization, while red boxes indicates missing or false positive in 3D localization.

tional efficiency of the proposed method allows it to handle such large-scale data more effectively, highlighting its scalability and practical applicability in real-world scenarios.

F. Broader Applications and Limitations

Broader application. The proposed method offers the potential for broader applications across diverse scenarios. Similar to works that explore the application in point cloud [4, 15] and MLP-based methods [18], our approach,

using 3DGS, can be extended to support various input modalities, such as click or image queries, by leveraging a self-referencing mechanism. Additionally, integrating our method with Large Language Models (LLMs) could facilitate dialogue-based interactions, allowing users to dynamically issue commands or explore the environment. This integration suggests promising avenues for developing 3D interactive systems that go beyond simple search tasks. Our approach is feature-agnostic, enabling straightforward extension to various feature representations, such as DINO [14]

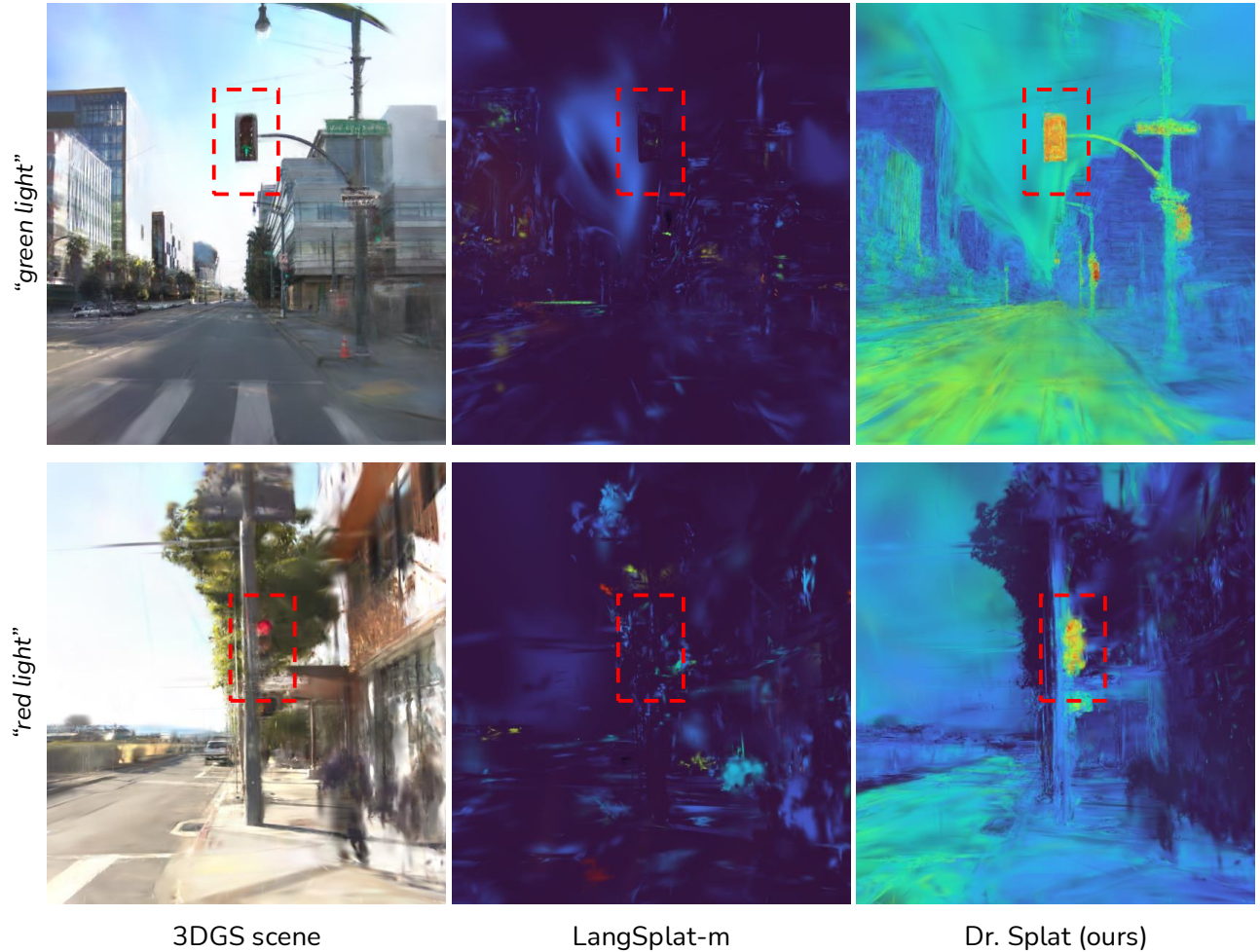


Figure S8. We compare 3D localization between rendering based Langsplat-m with registration based Dr. Splat. While LangSplat-m shows randomly distributed activations, fail to localize the target, ours model successfully detect the target in both cases.

or other features, allowing adaption to diverse applications, *i.e.* rendering stylization [6, 9, 10].

Furthermore, applying the method to canonical forms could support dynamic 3D scenes [13, 23]. This adaptation would extend the applicability of our approach beyond static environments, demonstrating its versatility in handling complex, real-world scenarios.

Limitation. While our method has demonstrated robust performance across diverse combinations of nouns and adjectives (*e.g.*, “tea in a glass,” wavy noodles,” and red light” in Fig. S6 and Fig. S10) as well as unfamiliar nouns (*e.g.*, nori,” waldo,” and safety cone”), without additional training, generalization remains an area for improvement. Exploring additional training techniques for Product Quantization (PQ) could further enhance the method’s capabilities. Further exploration of Product Quantization (PQ) training, such as using more diverse datasets or finer-grained query representations, could enhance adaptability across varied contexts.

Despite its advantages, some limitations of the proposed method have also been identified, particularly related to CLIP features. Occasionally, related but distinct objects are simultaneously activated for a given query. For instance, the query “red apple” might activate non-red apples or unrelated red objects. This stems from CLIP’s semantic associations and could be mitigated with post-processing techniques like re-ranking to improve query specificity.

Lastly, similar to previous methods [16, 20, 22], ours also requires to set an appropriate threshold. In this study, we utilize a fixed similarity threshold employed a fixed similarity threshold across all scenes, ensuring stable and reproducible results. However, optimizing thresholds for specific scenarios or implementing dynamic adjustments could further refine localization accuracy in diverse environments.

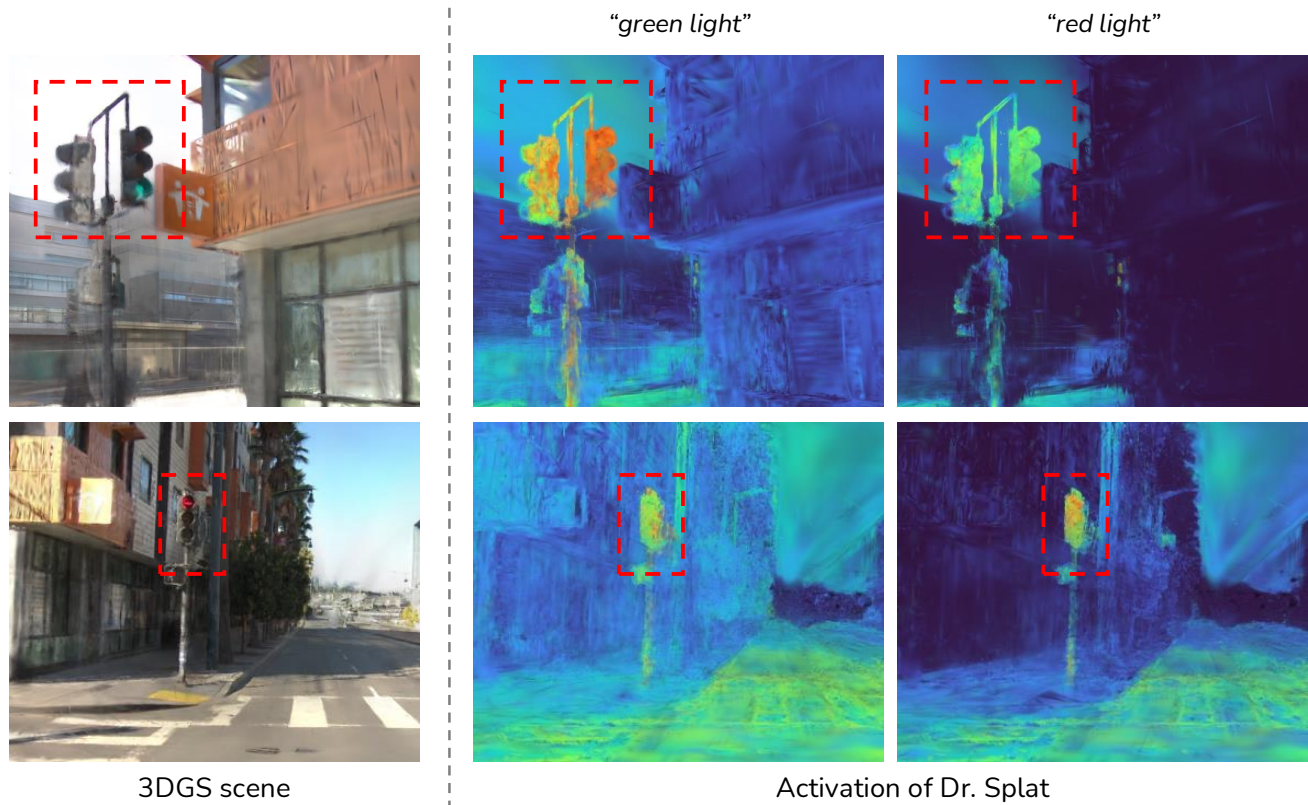


Figure S9. Visualization of 3d localization in different attributes (e.g., color) given as query. The result highlights the ability of Dr. Splat (ours) to effectively distinguish attributes such as “green light” and “red light” in scenes based on text queries, demonstrating the robustness in open-vocabulary understanding.



Figure S10. Qualitative results of Dr. Splat on 3D localization task in city-scale data showcasing Dr. Splat’s generalization performance across diverse text queries includes various target objects and concepts.

References

- [1] Alexandr Andoni, Piotr Indyk, Thijs Laarhoven, Ilya Razenshteyn, and Ludwig Schmidt. Practical and optimal lsh for angular distance. In *NeurIPS*, 2015. 6
- [2] 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. 3, 4, 5
- [3] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. *arXiv preprint arXiv:2401.08281*, 2024. 1
- [4] 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. 8
- [5] 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
- [6] Kim Jun-Seong, Kim Yu-Ji, Moon Ye-Bin, and Tae-Hyun Oh. Hdr-plenoxels: Self-calibrating high dynamic range radiance fields. In *ECCV*, 2022. 9
- [7] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM TOG*, 2023. 1, 3, 4
- [8] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lrf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19729–19739, 2023. 2, 3
- [9] GeonU Kim, Kim Youwang, and Tae-Hyun Oh. FPRF: Feed-forward photorealistic style transfer of large-scale 3D neural radiance fields. In *AAAI*, 2024. 6, 9
- [10] GeonU Kim, Kim Youwang, Lee Hyoseok, and Tae-Hyun Oh. Fpgs: Feed-forward semantic-aware photorealistic style transfer of large-scale gaussian splatting, 2025. 9
- [11] Jun Seong Kim, Mingyu Kim, GeonU Kim, Tae Hyun Oh, and Jin Hwa Kim. Factorized multi-resolution hashgrid for efficient neural radiance fields: Execution on edge-devices. In *IEEE Robotics and Automation Letters*, 2024. 6
- [12] 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. 2
- [13] Zhicheng Lu, Xiang Guo, Le Hui, Tianrui Chen, Ming Yang, Xiao Tang, Feng Zhu, and Yuchao Dai. 3d geometry-aware deformable gaussian splatting for dynamic view synthesis. In *CVPR*, 2024. 9
- [14] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 8
- [15] 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. 8
- [16] 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. 2, 3, 6, 7, 8, 9
- [17] 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
- [18] 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. 8
- [19] David Rozenberszki, Or Litany, and Angela Dai. Language-grounded indoor 3d semantic segmentation in the wild. In *ECCV*, 2022. 6
- [20] 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, 3, 6, 8, 9
- [21] Matthew Tancik, Vincent Casser, Xincheng 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*, pages 8248–8258, 2022. 6
- [22] Yanmin Wu, Jiarui Meng, Haijie Li, Chenming Wu, Yahao Shi, Xinhua Cheng, Chen Zhao, Haocheng Feng, Errui Ding, Jingdong Wang, et al. Opengaussian: Towards point-level 3d gaussian-based open vocabulary understanding. *arXiv preprint arXiv:2406.02058*, 2024. 2, 3, 5, 6, 7, 8, 9
- [23] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *CVPR*, 2024. 9