

# Semantic Consistent Language Gaussian Splatting for Point-Level Open-vocabulary Querying

Hairong Yin<sup>1</sup> Huangying Zhan<sup>2</sup> Yi Xu<sup>2</sup> Raymond A. Yeh<sup>1</sup>

<sup>1</sup>Department Computer Science, Purdue University

{yin178, rayyeh}@purdue.edu

{huangying.zhan, yi.xu}@goertekusa.com

## Abstract

*Open-vocabulary querying in 3D Gaussian Splatting aims to identify semantically relevant regions within a 3D Gaussian representation based on a given text query. Prior work, such as LangSplat, addressed this task by retrieving these regions in the form of segmentation masks on 2D renderings. More recently, OpenGaussian introduced point-level querying, which directly selects a subset of 3D Gaussians. In this work, we propose a point-level querying method that builds upon LangSplat’s framework. Our approach improves the framework in two key ways: (a) we leverage masklets from the Segment Anything Model 2 (SAM2) to establish semantic consistent ground-truth for distilling the language Gaussians; (b) we introduce a novel two-step querying approach that first retrieves the distilled ground-truth and subsequently uses the ground-truth to query the individual Gaussians. Experimental evaluations on three benchmark datasets demonstrate that the proposed method achieves better performance compared to state-of-the-art approaches. For instance, our method achieves an mIoU improvement of +20.42 on the 3D-OVS dataset. Project page see: <https://evelinyin.github.io/seconGS/>*

## 1. Introduction

Open-vocabulary 3D scene understanding is the task of interpreting a 3D scene by associating its components with natural language. A system that understands 3D scenes leads to many potential applications in robotics, graphics editing, human-computer interaction, etc. This task requires accurately interpreting complex 3D geometries and aligning them with language descriptions.

Recently, NeRF [25, 26, 41, 43] and 3D Gaussians [4, 15, 46] have emerged as popular 3D scene representations due to their high visual qualities and fast rendering speeds. This led to considerable interest [6, 23, 28, 35, 40, 52] in building

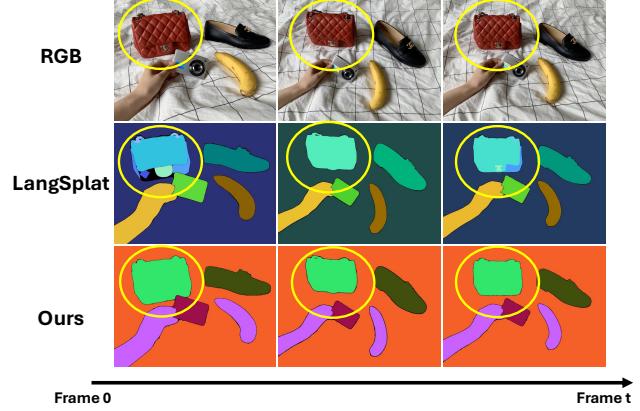


Figure 1. Visualization of the language embedding supervision. For the “red bag” circled in yellow, the ground-truth constructed by LangSplat [28] is inconsistent across frames while Ours remains consistent across frames.

open-vocabulary understanding capabilities on top of these 3D scene representations. With advancements in foundation 2D vision-language models (VLMs) [21, 29] and 3D-aware models [16], recent works [28, 35, 51] have further explored this area by incorporating language understanding into 3D representations through the introduction of language parameters. To evaluate open-vocabulary capabilities, these works have formulated the task of open-vocabulary querying.

Given a text query and a language-augmented 3D scene representation, LangSplat [28] formulates the task as retrieving the pixels related to the query, represented as a 2D mask. In this case, language features are rendered into an image, and the querying procedure involves comparing the query vector with the language feature at each pixel to identify relevant regions. However, 2D querying is limited to capturing 3D spatial information accurately, making it inadequate for tasks requiring precise 3D understanding. This leads to the focus on *direct querying in 3D* [22, 40, 44], i.e., point-level querying, where they identify relevant 3D parameters directly related to a text query.

For 3D Gaussian Splatting, the 3D querying task in-

Part of this work was done at OPPO US Research Center.

volves retrieving the relevant Gaussians. The primary advantage of direct querying in 3D is that associations are made directly to the 3D representation rather than through a rendered image. This could be beneficial for tasks that require direct access to 3D data, *e.g.*, robotics tasks [12, 34, 38], object editing [19, 39, 42], object selection [31], etc.

In this work, we propose a method, based on LangSplat’s framework, that leverages the power of Segment Anything Models [18, 30] to construct consistent language embeddings across different views. This consistency ensures robust supervision for training the language Gaussians, where the same embedding would be used to supervise the same object; See Fig. 1 for comparison with LangSplat. Furthermore, we introduce a two-step querying process to mitigate the challenge of selecting an effective similarity threshold between the text query embedding and the learned language embedding in LangSplat.

Empirically, we evaluate our method on three datasets: LERF [16], 3D-OVS [23], and Replica [37], demonstrating that it outperforms the state-of-the-art method in terms of mIoU by +4.16, +20.42, and +1.74, respectively.

#### Our main contributions are as follows:

- We introduce a method for generating semantic and 3D-consistent ground-truth to train language-aware Gaussians.
- With this improved 3D language Gaussians, we propose an effective two-step querying process by leveraging the created consistent ground-truth.
- Extensive experiments across multiple datasets demonstrate that our approach outperforms existing open-vocabulary 3D querying methods.

## 2. Related Work

**Vision foundation models.** In computer vision, several open-sourced foundation models have become the bedrock of many works. These foundation models’ capabilities can either be used directly or their features can be distilled into another model. In language and vision, CLIP [29] is a model that is capable of encoding images and natural language text to the same embedding space. Using this joint embedding space, they demonstrate zero-shot capability for image classification, which later is generalized to segmentation [50] and language segmentation (LSeg) [21].

In the area of image segmentation, the Segment Anything Model (SAM) [18] is a notable foundation model. SAM’s segmentation capabilities have been adapted and extended to 3D tasks. Recent works [3, 17] have leveraged SAM to integrate semantic information into NeRFs, enabling the extraction of 3D masks for target objects. Other approaches [11, 24] have incorporated semantic features into point clouds, enhancing object representation and segmentation in 3D. The process in 3D Gaussian Splatting [15] has further motivated studies [2, 9, 14, 33, 45, 47] focus on

object representation in both 3D and 4D, including advancements in interactive segmentation and object tracking.

Recently, SAM2 [30] extended SAM’s capability to tracking of masklets, *i.e.*, consistent masks across both space and time. In this work, we leverage masklets extracted by SAM2 to provide consistent guidance for training the language Gaussians.

**Open-vocabulary 3D scene understanding.** With advancements in 3D scene representation, there is a surge in interest in incorporating semantics/language into 3D representation. One line of work [6, 16, 23, 34, 43] distilled features from DINO [1], LatentDiffusion [32], LSeg [21], CLIP [29] to learn a NeRF [26], or leveraged 2D annotations [36, 49] to construct feature/language fields. Others [7, 13, 27, 48] distills knowledge from these language-rich models into point clouds, enabling open-vocabulary 3D scene understanding. In more recent works [10, 28, 35, 40, 51], there is a shift towards 3D Gaussian Splatting [15].

More closely related to our work is LangSplat [28], which augments 3D scenes with language features distilled from CLIP, enabling natural language querying on the renderings of the 3D scene. Gaussian Grouping [44] jointly performs 3D reconstruction and segmentation of open-world objects, leveraging SAM-based 2D mask predictions and 3D spatial constraints. However, they do not support open-vocabulary 3D querying. OpenGaussian [40] introduces new loss functions that leverage inter- and intra-mask smoothness relationships, along with a codebook-based clustering method to improve instance-level association of 3D points. Differently, our approach does not rely on new loss functions. Instead, we extend SAM by using SAM2’s extracted masklets to create consistent ground-truth supervision. Additionally, we propose a novel two-step querying procedure. These aspects have not been explored previously.

## 3. Preliminaries

We review LangSplat and Segment Anything Models (SAM) to introduce the necessary notation to understand our approach.

**LangSplat** [28] represents a 3D scene with a set of 3D Gaussians  $\mathcal{G} = \{g_i\}$ , where each Gaussian  $g_i$  is associated with the parameters

$$g_i \triangleq (\mu_i, \Sigma_i, c_i, \alpha_i, l_i) \quad (1)$$

corresponding to the 3D location, covariance matrix, color, opacity, and a language embedding. Different from a regular 3D Gaussian splatting [15], each of the Gaussians (Eq. (1)) includes a language embedding  $l_i \in \mathbb{R}^D$  to encode the semantics of a 3D scene.

This language embedding can then be rendered into a language field  $\hat{L}_\pi \in \mathbb{R}^{H \times W \times D}$ , where  $H$  and  $W$  correspond to the height and width of the rendered image at a

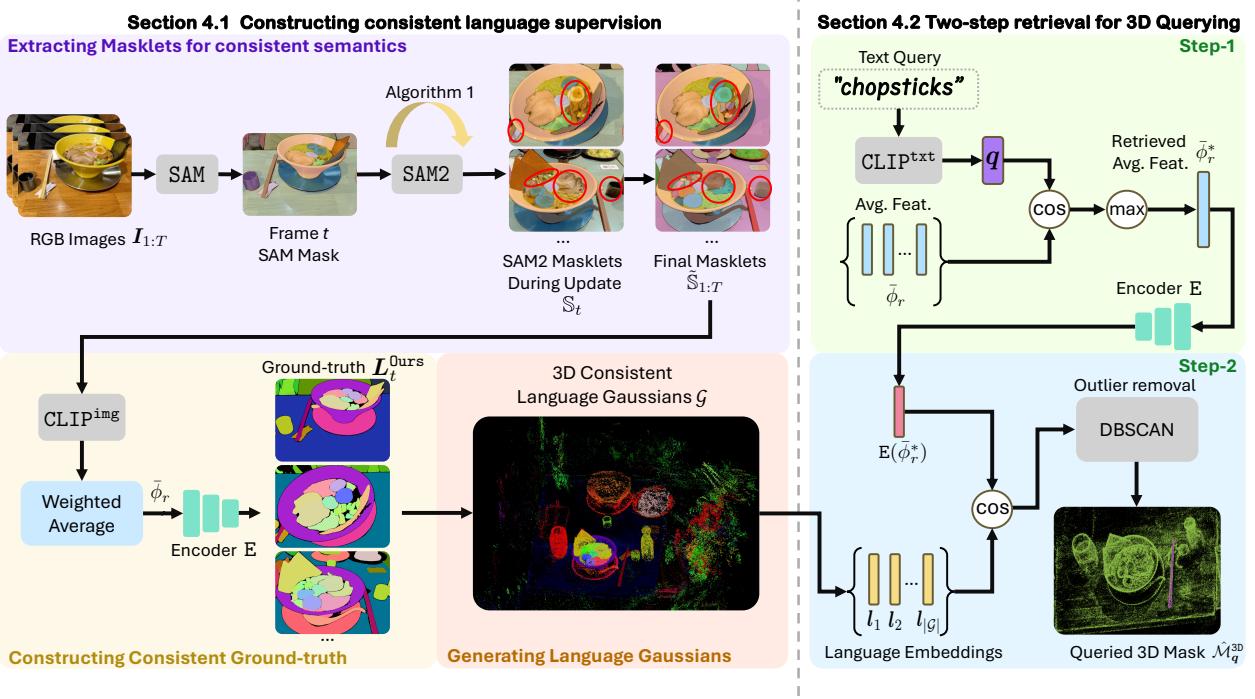


Figure 2. Overview of the proposed method. In Sec. 4.1, we present a masklet extraction algorithm (Alg. 1) that leverages Segment Anything Models to generate consistent ground-truth  $L_t^{\text{Ours}}$  for training the language parameters  $l_1, \dots, l_{|\mathcal{G}|}$ . In Sec. 4.2, we discuss the two-step retrieval procedure. Rather than directly querying the language parameters  $l_i$  with the query vector  $q$ , we first retrieve the features  $\bar{\phi}_r$  that are used to construct the ground truth  $L_t^{\text{Ours}}$ . In Step-2, we query the 3D language Gaussian using the encoded feature,  $E(\bar{\phi}_r^*)$ , obtained from Step-1.

camera pose  $\pi$ . This is done by using a tile-based rasterization [53], just as one would for colors. The language feature at each pixel  $p$  is computed as

$$\hat{L}_\pi[p] = \sum_{i \in \mathcal{T}} l_i f_i^{2D}(p) \prod_{j=1}^{i-1} (1 - f_j^{2D}(p)), \quad (2)$$

where  $f_i^{2D}$  represents the the projected 2D contribution of a 3D Gaussian  $g_i$ , and  $\mathcal{T}$  is the set of Gaussians in a tile.

**LangSplat training.** Let  $\hat{L}_{\pi_t}$  denote the rendering of the scene from the camera pose  $\pi_t$  associated with image  $I_t$ . LangSplat trains language embedding  $l_i$  by minimizing the L1 loss between the rendering of the language feature  $\hat{L}_\pi$  and a “ground-truth” language feature  $L_t$ , i.e.,

$$\min_{\{l_i \forall i\}} \mathbb{E}_{I_t} \left[ \text{L1}(\hat{L}_{\pi_t}, L_t) \right]. \quad (3)$$

Importantly, how one designs this “ground truth” significantly affects the query performance.

In LangSplat, the ground-truth is distilled from a pre-trained CLIP [5] with the help of the segment anything model (SAM) [18]. Given an RGB image  $I$ , SAM extracts a set of non-overlapping segmentation masks  $\mathbb{S}_I \triangleq \{\mathcal{S}_r\} = \text{SAM}(I)$ , where each  $\mathcal{S}_r$  corresponds to the segmented mask of a region  $r$ . Here, we broadly use the term “region” to

mean a set of related pixels, e.g., it could be an object or a subpart of an object.

Next, a CLIP embedding  $\phi_r$  is extracted for each region, LangSplat [28] masks the image, then passes it to CLIP’s image encoder  $\text{CLIP}^{\text{img}} : \mathbb{R}^{H \times W \times 3} \rightarrow \mathbb{R}^D$ , i.e.,

$$\phi_{r,t} \triangleq \text{CLIP}^{\text{img}}(I_t \odot S_r), \quad (4)$$

where  $S_r$  corresponds to the mask  $\mathcal{S}_r$  in matrix form, and  $\odot$  denotes an element-wise multiplication. These extracted region features are then placed back to their respective regions to form the ground-truth

$$L_t[(h, w)] \triangleq \phi_{r,t} \text{ if } (h, w) \in \mathcal{S}_r \forall r. \quad (5)$$

As the language embedding  $L$  is distilled from a pre-trained CLIP embedding, the query vector  $q$  is extracted from the pre-trained CLIP text encoder. This ensures that both the query vector and the 3D language embeddings are in the same space, allowing for retrieval.

Lastly, we note an implementation detail, LangSplat [28] trains an autoencoder, consisting of an encoder  $E$  and a decoder  $D$ , to reduce the dimensions of the CLIP features, where  $E \circ D$  approximates an identity function. With this autoencoder, all the aforementioned formulations can be done in a lower-dimensional space to save GPU memory.

**Open-Vocabulary 2D Querying.** Given the language embeddings, LangSplat [28] considers the task of *open vocabulary 2D querying* defined as finding the *set of pixels* that are relevant to the text query rendered at the given camera pose. Formally, given a query language vector  $\mathbf{q}$ , a trained LangSplat model  $\mathcal{G}$ , and a camera  $\pi$ , the goal is to retrieve a *2D mask*

$$\hat{\mathcal{M}}_{\pi, \mathbf{q}}^{2D} \subseteq \{1 \dots H\} \times \{1 \dots W\}, \quad (6)$$

represented in a set notation, where related pixels are within the set. This set notation can also be represented as a matrix

$$\hat{\mathbf{M}}_{\pi, \mathbf{q}}^{2D}[h, w] = \begin{cases} 1, & \text{if } (h, w) \in \hat{\mathcal{M}}_{\pi, \mathbf{q}}^{2D} \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

In LangSplat, the mask  $\hat{\mathcal{M}}_{\pi, \mathbf{q}}^{2D}$  is created by thresholding the similarity between the query  $\mathbf{q}$  and the rendered language features  $\hat{\mathbf{L}}_\pi$  at each pixel location, *i.e.*,

$$\hat{\mathcal{M}}_{\pi, \mathbf{q}}^{2D} \triangleq \{(h, w) | \text{Sim}(\hat{\mathbf{L}}_\pi[h, w], \mathbf{q}) \geq C \forall h, w\}, \quad (8)$$

where  $C$  denotes the threshold and  $\text{Sim}$  denotes the similarity metric.

**Open-vocabulary 3D (point-level) querying.** In comparison, OpenGaussian [40] proposes to directly query the 3D Gaussians with natural language. Formally, given a trained 3D scene  $\mathcal{G}$  and a query  $\mathbf{q}$ , the task is to predict a “*3D mask*”

$$\mathcal{M}^{3D} = \{\mathbf{g}_i\} \subseteq \mathcal{G}, \quad (9)$$

that indicates whether each Gaussian  $\mathbf{g}_i$  is relevant to a text query  $\mathbf{q}$ . In other words, a *point-level* querying on the 3D scene’s representation rather than on a *rendered image* of a 3D scene.

**Segment Anything Model 2 (SAM2)** [30]. Given a sequence of frames  $\mathbf{I}_{t_1:t_2}$ , and a set of boxes  $\tilde{\mathcal{B}}_{t_1}$  indicating the regions in frame  $t_1$  that we wish to track and segment, SAM2 extracts a set  $\tilde{\mathcal{S}}_{\mathbf{I}_{t_1:t_2}}$  of non-overlapping masks across both *space and time*, *i.e.*, “masklets”

$$\tilde{\mathcal{S}}_{\mathbf{I}_{t_1:t_2}} \triangleq \{\tilde{\mathcal{S}}_r\} = \text{SAM2}(\mathbf{I}_{t_1:t_2}, \mathcal{B}_{t_1}), \quad (10)$$

where each  $\tilde{\mathcal{S}}_r$  is a set containing voxels  $(t, h, w)$  that are associated with the region  $r$ . Following the same syntax for a 2D binary mask, we use the tensor  $\tilde{\mathcal{S}}_r \in \{0, 1\}^{T \times H \times W}$  to represent the masklet  $\tilde{\mathcal{S}}_r$  in set notation.

## 4. Towards Open-Vocabulary 3D Querying

Following LangSplat’s framework, we use a set of 3D Gaussians augmented with language embeddings to represent a scene. Differently, we propose a novel method for constructing ground-truths that are more semantically consistent and robust across various 3D viewpoints (Sec. 4.1). This approach helps the training of better language embeddings for querying. We then introduce a querying method tailored for our learned embeddings (Sec. 4.2). See Fig. 2 for a visual overview of our method.

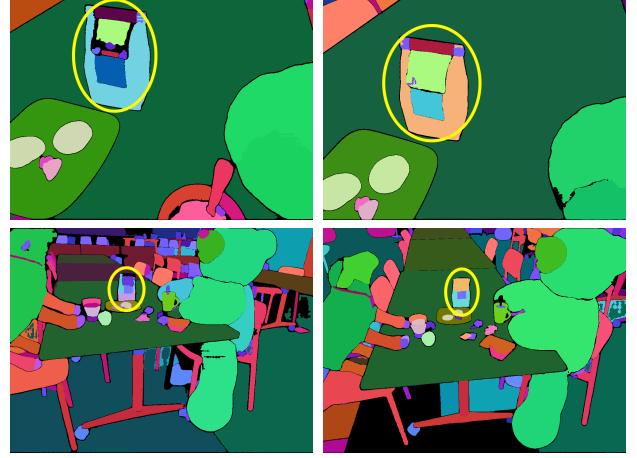


Figure 3. Visualization of the ground-truth  $L_t$  constructed by baseline LangSplat [28]. As can be observed, some semantics are not consistent across viewpoints, *e.g.*, the circled bag of coffee.

### 4.1. Constructing consistent language supervision

Given a sequence of frames  $[\mathbf{I}_1, \dots, \mathbf{I}_T]$  with camera poses, we aim to construct a better ground-truth feature  $L_t^{\text{Ours}}$  for each of the frames to train LangSplat’s parameters by minimizing the objective function in Eq. (3). This improved ground-truth is obtained by ensuring that all pixels within the same region, as identified by SAM2, share the same CLIP embedding. In other words, the supervision will be semantically consistent up to the quality of the extracted masklets from SAM2.

**Inconsistent semantics from LangSplat.** The main shortcoming of ground-truth feature  $L_t$  created by LangSplat is its potential inconsistency across different views. In Fig. 3, we visualize the ground-truth feature constructed by LangSplat, where similar colors indicate more similar features. As shown, the bag of cookies (circled in yellow) exhibits significant color variation across the views, indicating that the supervision provided by LangSplat is inconsistent. This inconsistency arises because SAM segments each individual image independently, potentially selecting different regions to segment. Hence, the ground-truth features derived from these segmentations are also inconsistent.

To address this inconsistency of LangSplat, we construct the ground truth  $L_t^{\text{Ours}}$  by additionally leveraging the segmentation and tracking capabilities of SAM2. Importantly, we aim to design  $L_t^{\text{Ours}} \in \mathbb{R}^{H \times W \times D}$  such that each pixel’s feature remains consistent across frames when it belongs to the same region extracted by SAM2.

**Extracting masklets for consistent semantics.** As reviewed in Sec. 3, SAM2 takes a sequence of images and bounding boxes as input to track masks of the same region. In contrast, SAM takes a single image and proposes regions for segmentation at different levels but does not perform tracking. To combine the best of both worlds, we use SAM

---

**Algorithm 1** Extracting regions with SAM and SAM2

---

```

1: Input: Image sequence  $\mathbf{I}_{1:T}$ , segmentors SAM and
   SAM2, threshold  $\kappa$ 
2:  $\tilde{\mathcal{S}}_{1:T} \leftarrow \{\} \# \text{ Tracked masklets}$ 
3: for  $t \in \{1, \dots, T\}$  do
4:    $\mathbb{S}_{\mathbf{I}_t} = \text{SAM}(\mathbf{I}_t)$ 
5:   # Check if an region is tracked.
6:   for  $S_r \in \mathbb{S}_{\mathbf{I}_t}$  do
7:     for  $\tilde{S}_{r'} \in \tilde{\mathcal{S}}_{1:T}$  do
8:       if  $\text{IoU}(\tilde{S}_{r'}[t-1], S_r) > \kappa$  then
9:          $\mathbb{S}_{\mathbf{I}_t} \leftarrow \mathbb{S}_{\mathbf{I}_t} \setminus \{S_r\}$ 
10:        end if
11:      end for
12:    end for
13:    # Adding untracked masklets
14:     $\mathcal{B} \leftarrow \text{GetBoxFromMask}(\mathbb{S}_{\mathbf{I}_t})$ 
15:     $\tilde{\mathcal{S}}_{1:T} \leftarrow \tilde{\mathcal{S}}_{1:T} \cup \text{SAM2}(\mathbf{I}_{1:T}, \mathcal{B})$ 
16:  end for
17: Output:  $\tilde{\mathcal{S}}_{1:T}$ 

```

---

to propose regions and SAM2 to track and segment them, resulting in a set of masklets  $\tilde{\mathcal{S}}_{1:T}$ .

For each frame  $\mathbf{I}_t$ , SAM proposes a set of regions  $\mathbb{S}_{\mathbf{I}_t}$ , where bounding boxes are extracted for SAM2. Specifically, we take the top-left and bottom-right corner pixels of each region to form a box. Next, we check if a proposed region has already been tracked by SAM2 by comparing the mIoU of the SAM mask with the tracked masks and applying a threshold. If the proposed region has not been tracked, we run SAM2 using its bounding box and add the output masklets to the set of tracked masklets  $\tilde{\mathcal{S}}_{1:T}$ . These steps are summarized in Alg. 1.

**Constructing consistent ground-truth.** With the set of masklets  $\tilde{\mathcal{S}}_{1:T}$  extracted, we create an average CLIP embedding  $\bar{\phi}_r$  for each masklet  $\tilde{S}_r \in \tilde{\mathcal{S}}_{1:T}$ . This is done by masking out the image  $\mathbf{I}_t$  using the extracted masklet and then passing it to CLIP’s image encoder, *i.e.*,

$$\bar{\phi}_r = \sum_{t=1}^T \omega_t \cdot \text{CLIP}^{\text{img}}(\mathbf{I}_t \odot \tilde{S}_r[t]), \quad (11)$$

where  $\tilde{S}_r$  denotes the masklet  $\tilde{S}_r$  represented in a tensor, and  $\omega_t$  denotes the ratio of pixels in  $\tilde{S}_r[t]$  to the total pixel count in  $\tilde{S}_r$ . In other words, the embedding from each view is weighted proportionally to the number of pixels in the segmentation.

Intuitively, as averaging reduces variance, the proposed  $\bar{\phi}_r$  is more consistent than the individual  $\phi_{r,t}$  used in LangSplat. Furthermore, the weighting scheme helps to reduce the influence of small regions that often contain noisier language embeddings. In contrast, a naive equal weight to each observation from different views disregards the reliability of individual features.

To construct the ground truth  $\mathbf{L}_t^{\text{Ours}}$  for a frame  $\mathbf{I}_t$ , we place the averaged CLIP embedding into the pixel location of each frame according to the masklet. For all extracted masklets  $\tilde{S}_r \in \tilde{\mathcal{S}}_{1:T}$ ,

$$\mathbf{L}_t^{\text{Ours}}[(h, w)] = \bar{\phi}_r \text{ if } (t, h, w) \in \tilde{S}_r. \quad (12)$$

As shown, for pixels across views that are tracked and segmented into the same region, this construction of ground-truth assigns the same averaged CLIP embedding. Compared with LangSplat’s ground-truth in Eq. (5), our construction is shared across time  $t$ , *i.e.*, the language embedding  $\mathbf{l}_i$  in LangSplat receives consistent supervision across all relevant frames.

**Autoencoder details.** As in LangSplat, to reduce the GPU memory usage, we train a light-weight autoencoder consisting of an encoder  $E$  and a decoder  $D$ . Differently, we encode the averaged CLIP embedding  $\bar{\phi}_r$  into a low-dimensional latent space, *i.e.*,  $E(\bar{\phi}_r)$  is used as supervision to train the language embeddings  $\mathbf{l}_i$ .

## 4.2. Two-step retrieval for 3D Querying

With the text query vector  $\mathbf{q}$ , the standard one-step approach directly compares the CLIP features  $\mathbf{q}$  of the query text with the language embeddings  $\mathbf{l}_i$  of each Gaussian, *i.e.*

$$\hat{\mathcal{M}}_{\mathbf{q}}^{\text{3D,1step}} \triangleq \{\mathbf{g}_i \mid \forall i \cos(\mathbf{l}_i, \mathbf{q}) \geq \text{threshold}\}, \quad (13)$$

where  $\cos$  is the cosine similarity. However, this one-step approach struggles to find a single effective threshold across different language embedding  $\mathbf{l}_i$  as CLIP’s image and language embedding are known to be not well calibrated [20]. To address this challenge, we propose our two-step approach for querying 3D Gaussians, specifically designed to mitigate this problem.

**Step-1.** Given the CLIP feature of a text query  $\mathbf{q} \in \mathbb{R}^{512}$ , we first retrieve the most similar average feature over all regions feature

$$\bar{\phi}_r^* \triangleq \arg \max_{\bar{\phi}_r} \cos(\bar{\phi}_r, \mathbf{q}). \quad (14)$$

Since  $\bar{\phi}_r$  is obtained as a weighted average of CLIP image embeddings and  $\mathbf{q}$  comes from CLIP text embeddings, a direct comparison between them through cosine similarity is effective and does not involve a threshold.

**Step-2.** With the retrieved average feature  $\bar{\phi}_r^*$ , we leverage the pretrained encoder  $E$  to compress it into lower dimension. Then we compute its cosine similarity with the learned language embedding  $\mathbf{l}_i$  for each Gaussian. Next, we threshold this similarity to retrieve the tentative set of Gaussians

$$\tilde{\mathcal{M}}_{\mathbf{q}}^{\text{3D}} = \{\mathbf{g}_i \mid \forall i \cos(\mathbf{l}_i, E(\bar{\phi}_r^*)) \geq \text{threshold}\}, \quad (15)$$

where the querying process compares  $E(\bar{\phi}_r^*)$  with  $\mathbf{l}_i$ . Recall  $E(\bar{\phi}_r^*)$  is used as supervision for training language embeddings  $\mathbf{l}_i$ , *i.e.*, the relevant language embeddings are trained

Methods	mIoU↑					mAcc↑					Loc. Acc↑				
	figurines	ramen	teatime	kitchen	Avg	figurines	ramen	teatime	kitchen	Avg	figurines	ramen	teatime	kitchen	Avg
LangSplat [28]	12.43	6.39	20.60	17.58	14.25	21.43	7.04	37.29	18.18	20.99	5.36	0.00	3.39	4.55	3.33
OpenGauss. [40]	39.29	31.01	<b>60.44</b>	22.7	38.36	55.36	42.25	<b>76.27</b>	31.82	51.43	-	-	-	-	-
GSGroup. [44]	7.75	8.80	10.94	16.29	10.95	10.71	9.86	10.17	27.27	14.50	10.71	2.82	5.08	4.55	5.79
<b>Ours</b>	<b>58.98</b>	<b>37.85</b>	43.57	<b>29.67</b>	<b>42.52</b>	<b>82.14</b>	<b>61.97</b>	54.24	<b>50.00</b>	<b>62.09</b>	<b>30.36</b>	<b>9.86</b>	<b>11.86</b>	<b>13.64</b>	<b>16.43</b>

Table 1. Quantitative results on the LERF dataset. For OpenGaussian, we report the numbers from their paper.

Methods	mIoU↑					mAcc↑					Loc. Acc↑							
	bed	bench	lawn	room	Avg	bed	bench	lawn	room	Avg	bed	bench	lawn	room	Avg			
LangSplat [28]	29.83	17.38	33.64	23.35	24.64	25.77	43.33	28.57	63.33	33.33	43.33	42.38	3.00	48.57	40.00	43.33	43.33	35.64
OpenGauss. [40]	48.50	46.02	64.63	47.60	44.06	50.16	<b>100.00</b>	57.14	<b>100.00</b>	<b>83.33</b>	<b>66.67</b>	81.43	23.33	37.14	20.00	50.00	56.67	37.43
GSGroup. [44]	48.51	35.49	65.13	46.39	29.86	45.08	<b>100.00</b>	71.43	<b>100.00</b>	76.67	40.00	77.62	<b>96.67</b>	42.86	<b>100.00</b>	53.33	43.33	67.24
<b>Ours</b>	<b>56.81</b>	<b>87.58</b>	<b>87.12</b>	<b>64.70</b>	<b>56.70</b>	<b>70.58</b>	66.67	<b>100.00</b>	<b>100.00</b>	<b>83.33</b>	<b>66.67</b>	<b>83.33</b>	83.33	<b>100.00</b>	<b>100.00</b>	<b>83.33</b>	<b>100.00</b>	<b>93.33</b>

Table 2. Quantitative results on the 3D-OVS dataset.

to be similar to  $E(\bar{\phi}_r^*)$ . Therefore, any high threshold works well, which improves the queries’ reliability and robustness. More details are provided in the Appendix Sec. A4.

Finally, we apply DBSCAN [8] as post-processing to this retrieved set to remove outliers based on the spatial position  $\mu_i$  of the Gaussians.

## 5. Experiments

**Datasets.** Following LangSplat [28], we conduct experiments on the further annotated LERF [16] dataset that contains a set of in-the-wild scenes and on the 3D-OVS [23] dataset, which includes a collection of long-tail objects for evaluating open-vocabulary 3D querying. Additionally, we report results on the Replica [37] dataset, which has labeled point clouds for indoor objects. We randomly choose 10 classes as text queries from the 8 chosen scenes of Replica.

**Evaluation metrics.** Since the ground truths for LERF and 3D-OVS datasets are provided in 2D, we measure the performance of 3D querying indirectly. We render the queried Gaussians in  $\hat{\mathcal{M}}_q^{3D}$  to obtain a 2D mask for evaluation using 2D metrics following LangSplat [28]. This includes mean Intersection over Union (mIoU↑) and localization accuracy (Loc. Acc↑). Here, mIoU is defined as the ratio of the intersecting pixels to the total number of pixels in the union of the predicted masks and ground-truth masks. For Loc. Acc, a query is considered correct if the center of the queried mask’s exterior bounding box falls within the bounding box of the ground truth. We also report mIoU accuracy (mAcc↑), a 2D metric proposed by OpenGaussian [40], where a query is considered correct if its IoU is greater than 0.25. For the Replica dataset, which contains 3D retrieval ground truths, mIoU and mAcc are instead computed on the set of 3D locations of the Gaussians.

**Baselines.** We consider three baselines:

- *LangSplat* [28], which also trains a language 3D Gaussian. Hence, the 1-step query approach in Eq. (13) can be directly used, following how OpenGaussian [40] eval-

uates.

- *OpenGaussian* [40] proposed the task of open-vocabulary 3D (point-level) querying. As they reported on the LERF dataset, we directly included their results from the paper. For 3D-OVS and Replica datasets, we use their publicly released implementation.
- *GaussianGrouping* [44] We follow their implementation for the open-vocabulary querying based on the first frame’s semantics.

**Implementation details.** To extract language features, we use the OpenCLIP ViT-B/16 model. For 2D mask segmentation, we employ the SAM ViT-H model, and for tracking masks of the same object, we utilize the SAM2-hiera-large model. Other hyperparameters regarding 3D Gaussians follow LangSplat. More details are in the Appendix Sec. A3.

### 5.1. Quantitative results

**LERF dataset.** In Tab. 1, we show the results on the LERF dataset. We observe that Ours consistently outperforms LangSplat and, on average, is better than OpenGaussian, achieving an improvement of +4.16 in mIoU and +10.66 in mAcc. We observe that LangSplat and GaussianGrouping, methods designed for 2D querying, do not generalize well to point-level querying.

**3D-OVS dataset.** The results for the 3D-OVS dataset is reported in Tab. 2. Our method achieves 70.58% in mIoU and 93.33% in Loc. Acc, significantly outperforming baseline methods. Notably, our method demonstrates a +20.42 gain in mIoU averaged across scenes. We observe that there exists 100% in the Loc. Acc because the dataset is relatively easy with  $\leq 30$  frames and  $\leq 7$  objects in each scene, leading to effective segmentation and tracking from SAM2. Additionally, we observed that OpenGaussian has a low Loc. Acc due to small “floaters” in the retrieved Gaussians. These floaters shift the center point of the exterior bounding box to an incorrect location.

**Replica dataset.** We show experimental results in Tab. 3. We observe that our approach outperforms OpenGaussian

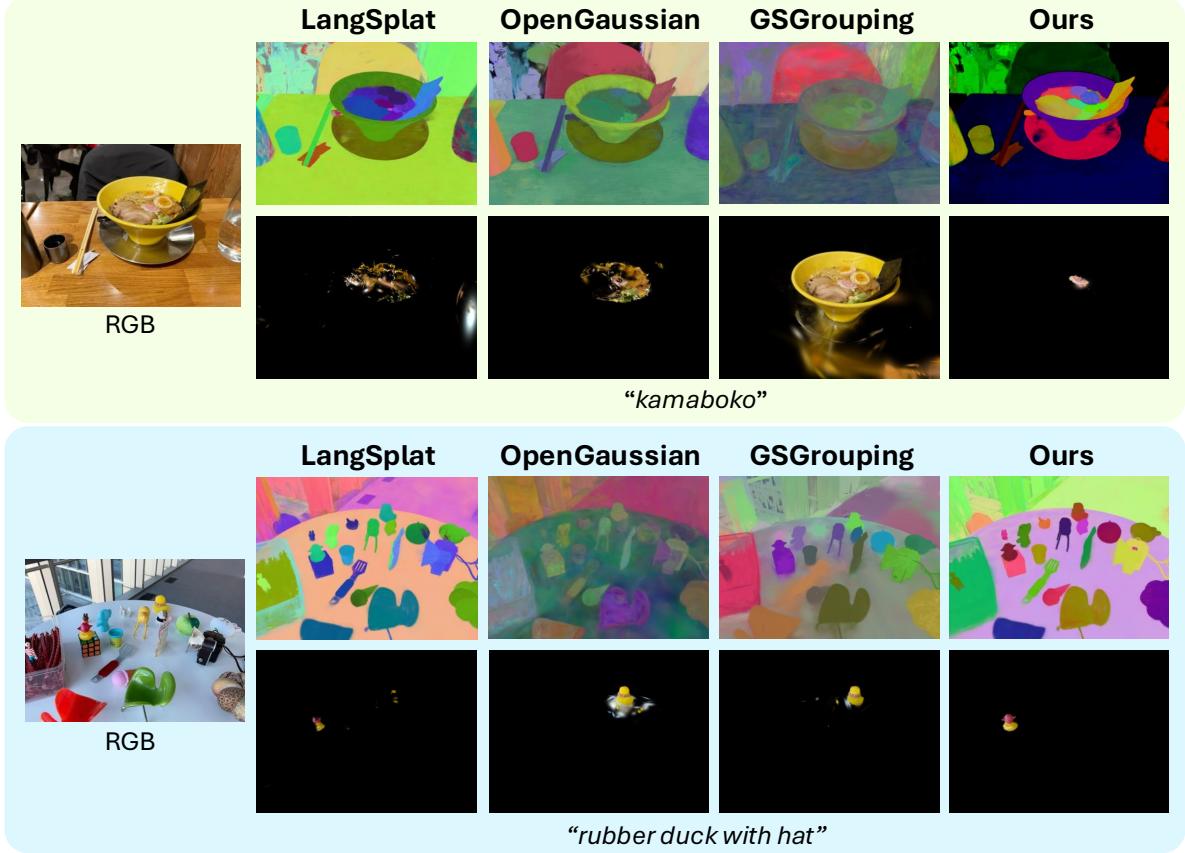


Figure 4. Qualitative results on LERF dataset of scene “ramen” and “figurines”. For each scene, the first row contains rendered language embeddings, and the second row contains 3D query results.

Methods	mIoU↑									mAcc↑								
	office0	office1	office2	office3	office4	room0	room1	room2	Avg	office0	office1	office2	office3	office4	room0	room1	room2	Avg
LangSplat [28]	2.43	2.1	5.68	4.65	1.49	3.86	4.08	0.92	3.15	11.09	1.36	10.7	13.99	2.37	12.82	12.24	10.05	9.33
OpenGauss. [40]	17.20	<b>23.13</b>	<b>43.72</b>	<b>42.36</b>	61.33	31.45	40.36	42.14	37.71	36.54	35.11	66.38	42.64	69.62	41.74	31.72	54.01	47.22
GSGroup. [44]	19.58	0.00	32.77	10.18	30.29	13.08	17.81	17.06	17.60	38.42	0.00	<b>74.48</b>	26.17	45.67	36.21	31.57	24.17	34.59
<b>Ours</b>	<b>25.77</b>	20.15	15.06	37.29	<b>64.83</b>	<b>40.33</b>	<b>64.39</b>	<b>47.81</b>	<b>39.45</b>	<b>50.76</b>	<b>35.97</b>	29.01	<b>45.12</b>	<b>82.85</b>	<b>60.00</b>	<b>84.72</b>	<b>63.64</b>	<b>56.51</b>

Table 3. Quantitative results on Replica dataset. Note, to compute mIoU and mAcc using the ground-truth point clouds, we skip the densification stage when training 3D Gaussian Splatting for all methods.

with an average gain of +1.74 on mIoU and +9.29 on mAcc. There exists 0.00 in GaussianGrouping’s results because their implementation only uses the first frame’s semantics. For large scenes such as Replica, the query object may not appear in the first frame, leading to an empty query.

## 5.2. Qualitative results

**LERF dataset.** To further analyze the methods, we visualize the learned language embeddings and the queried Gaussians in Fig. 4. We observe that baselines produce less consistent language embeddings, evident from the blurriness along object boundaries. Our method achieves cleaner and more fine-grained object localization in 3D space based on text queries, as illustrated in the second row for each scene in Fig. 4. Note that all four methods encounter a common

failure mode of empty query, *i.e.*, no valid Gaussians are returned for a text query, resulting in zero IoU for that query.

**3D-OVS dataset.** Fig. 5 shows the qualitative results on 3D-OVS dataset. Our method creates more consistent language embeddings with crisp object boundaries. Furthermore, our method’s retrieval results are less noisy and preserve the complete structure of each queried object, *i.e.*, the lawn is accurately retrieved with query “green lawn”.

**Replica dataset.** Fig. 6 visualizes the queried points for a scene in the Replica dataset given the query “cloth”. We observe that LangSplat and GaussianGrouping failed to retrieve the correct object, and OpenGaussian only retrieves part of the cloth with noisy points from other objects. Overall, our method retrieved a more complete and clearer ob-

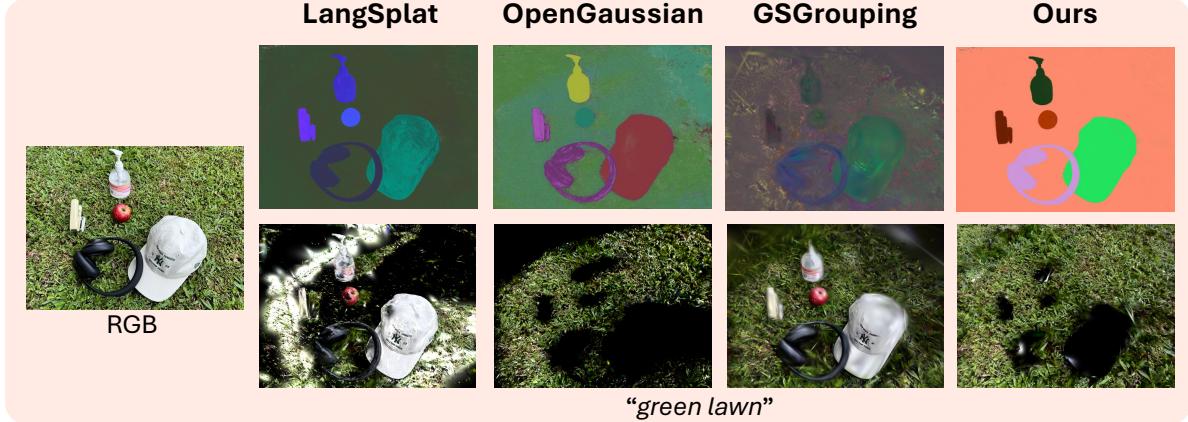


Figure 5. Qualitative results on 3D-OVS dataset for scene “lawn”. The first row contains rendered language embeddings, and the second row contains 3D query results for “green lawn”.

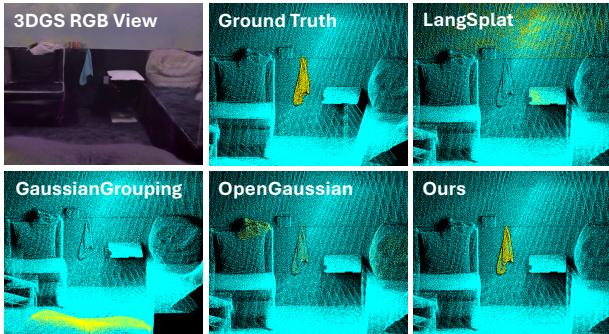


Figure 6. Qualitative results on Replica dataset. Yellow points are the queried points of “cloth”.

ject, which is consistent with the quantitative result.

### 5.3. Ablation study.

We conduct ablation studies to validate the efficacy of each proposed component of our method and report the performance in Tab. 4. Recall,  $\bar{\phi}_r$  denotes the *weighted* average features. Without SAM2’s masklets to get consistent features of the region, the performance drops significantly. Also, the 2-step query helps to get more accurate retrieval in 3D space. Overall, we observe that all proposed components are necessary to achieve the optimal performance.

We also conduct ablation studies on the effectiveness of our method without DBSCAN and evaluate the performance of canonical querying from LERF [16] in 3D querying. The original definition of canonical query from LERF is conducted on rendered 2D images. To evaluate the performance of canonical query in 3D querying, we replace  $\phi_{\text{lang}}$  by  $\mathbf{l}_i$  in the original equation. Formally, for each language embedding  $\mathbf{l}_i$  and each text query  $\mathbf{q}$ , the relevancy score is defined as  $\min_i \frac{\exp(\mathbf{l}_i \cdot \mathbf{q})}{\exp(\mathbf{l}_i \cdot \mathbf{q}) + \exp(\mathbf{q} \cdot \phi_{\text{canon}}^i)}$  where  $\phi_{\text{canon}}^i$  is the CLIP embedding of a predefined *canonical* phrase chosen from “object”, “things”, “stuff”, and “texture”.

SAM2	Ablations		Performance Metrics		
	$\bar{\phi}_r$	2-Step	mIoU	mAcc	Loc. Acc
-	✓	✓	4.56	7.14	5.36
✓	✓	✓	48.84	67.86	23.21
✓	✓	✓	9.09	12.50	8.93
✓	✓	✓	<b>58.98</b>	<b>82.14</b>	<b>30.36</b>

Table 4. Ablation study on each proposed component using LERF’s “figurines” scene. Note, the average feature  $\bar{\phi}_r$  cannot be implemented without SAM2. The best result is achieved with all our proposed components.

SAM2	Ablations			Metrics		
	2-Step	DBSCAN	Cano. Query	mIoU	mAcc	Loc. Acc
✓	✓	✓	✓	15.67	26.79	14.29
✓	✓	✓	✓	40.06	66.07	14.50
✓	✓	✓	✓	<b>58.98</b>	<b>82.14</b>	<b>30.36</b>

Table 5. Ablation study on DBSCAN and Canonical Query.

Next, we show ablations on canonical query in Tab. 5. We see that without DBSCAN, our proposed method also gains improvement in the performance. We also see that our proposed 2-step query significantly outperforms the canonical query.

## 6. Conclusion

We study the task of open-vocabulary 3D understanding formulated as a point-level 3D querying task. Based on LangSplat’s framework, we propose to improve the ground truth supervision to train more consistent language parameters by leveraging masklets extracted from SAM2. Furthermore, we introduce a novel 2-step querying pipeline to address the difficulty of choosing a consistent threshold in the baseline’s single-step query. Experiments over three datasets demonstrate that our approach achieves state-of-the-art performance.

## 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 *ICCV*, 2021. 2
- [2] Jiazhong Cen, Jiemin Fang, Chen Yang, Lingxi Xie, Xiaopeng Zhang, Wei Shen, and Qi Tian. Segment any 3d Gaussians. *arXiv preprint arXiv:2312.00860*, 2023. 2
- [3] Jiazhong Cen, Zanwei Zhou, Jiemin Fang, Wei Shen, Lingxi Xie, Dongsheng Jiang, Xiaopeng Zhang, Qi Tian, et al. Segment anything in 3d with NeRFs. In *NeurIPS*, 2023. 2
- [4] Guikun Chen and Wenguan Wang. A survey on 3D Gaussian splatting. *arXiv preprint arXiv:2401.03890*, 2024. 1
- [5] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Itsev. Reproducible scaling laws for contrastive language-image learning. In *CVPR*, 2023. 3
- [6] Zi-Ting Chou, Sheng-Yu Huang, I Liu, Yu-Chiang Frank Wang, et al. GSNeRF: Generalizable semantic neural radiance fields with enhanced 3d scene understanding. In *CVPR*, 2024. 1, 2
- [7] Runyu Ding, Jihan Yang, Chuhui Xue, Wenqing Zhang, Song Bai, and Xiaojuan Qi. PLA: Language-driven open-vocabulary 3d scene understanding. In *CVPR*, 2023. 2
- [8] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, 1996. 6
- [9] Qiao Gu, Zhaoyang Lv, Duncan Frost, Simon Green, Julian Straub, and Chris Sweeney. EgoLifter: Open-world 3d segmentation for egocentric perception. In *ECCV*, 2024. 2
- [10] 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
- [11] Shuting He, Henghui Ding, Xudong Jiang, and Bihan Wen. SegPoint: Segment any point cloud via large language model. In *ECCV*, 2024. 2
- [12] Yining Hong, Zishuo Zheng, Peihao Chen, Yian Wang, Junyan Li, and Chuang Gan. Multiply: A multisensory object-centric embodied large language model in 3d world. In *CVPR*, 2024. 2
- [13] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Osama, Tao Chen, Alaa Maalouf, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, et al. ConceptFusion: Open-set multimodal 3D mapping. *RSS*, 2023. 2
- [14] Shengxiang Ji, Guanjun Wu, Jiemin Fang, Jiazhong Cen, Taoran Yi, Wenyu Liu, Qi Tian, and Xinggang Wang. Segment any 4d Gaussians. *arXiv preprint arXiv:2407.04504*, 2024. 2
- [15] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM TOG*, 2023. 1, 2
- [16] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. LERF: Language embedded radiance fields. In *ICCV*, 2023. 1, 2, 6, 8
- [17] Chung Min Kim, Mingxuan Wu, Justin Kerr, Ken Goldberg, Matthew Tancik, and Angjoo Kanazawa. GARField: Group anything with radiance fields. In *CVPR*, 2024. 2
- [18] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, 2023. 2, 3
- [19] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing nerf for editing via feature field distillation. In *NeurIPS*, 2022. 2
- [20] Will LeVine, Benjamin Pikus, Pranav Raja, and Fernando Amat Gil. Enabling calibration in the zero-shot inference of large vision-language models. In *Proc. ICLR Workshop on Trustworthy ML*, 2023. 5
- [21] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and René Ranftl. Language-driven semantic segmentation. *ICLR*, 2022. 1, 2
- [22] Siyun Liang, Sen Wang, Kunyi Li, Michael Niemeyer, Stefano Gasperini, Nassir Navab, and Federico Tombari. SuperGSeg: Open-vocabulary 3D segmentation with structured super-gaussians. *arXiv preprint arXiv:2412.10231*, 2024. 1
- [23] Kunhao Liu, Fangneng Zhan, Jiahui Zhang, Muyu Xu, Yingchen Yu, Abdulkotaleb El Saddik, Christian Theobalt, Eric Xing, and Shijian Lu. Weakly supervised 3d open-vocabulary segmentation. In *NeurIPS*, 2023. 1, 2, 6
- [24] Youquan Liu, Lingdong Kong, Jun Cen, Runnan Chen, Wenwei Zhang, Liang Pan, Kai Chen, and Ziwei Liu. Segment any point cloud sequences by distilling vision foundation models. In *NeurIPS*, 2024. 2
- [25] 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 *CVPR*, 2021. 1
- [26] 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*, 2021. 1, 2
- [27] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. OpenScene: 3D scene understanding with open vocabularies. In *CVPR*, 2023. 2
- [28] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language Gaussian splatting. In *CVPR*, 2024. 1, 2, 3, 4, 6, 7
- [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 *ICML*, 2021. 1, 2
- [30] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chaoyuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. In *ICLR*, 2025. 2, 4

- [31] Zhongzheng Ren, Aseem Agarwala, Bryan Russell, Alexander G Schwing, and Oliver Wang. Neural volumetric object selection. In *CVPR*, 2022. 2
- [32] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 2
- [33] QiuHong Shen, Xingyi Yang, and Xinchao Wang. FlashSplat: 2d to 3D Gaussian splatting segmentation solved optimally. In *ECCV*, 2024. 2
- [34] William Shen, Ge Yang, Alan Yu, Jansen Wong, Leslie Pack Kaelbling, and Phillip Isola. Distilled feature fields enable few-shot language-guided manipulation. In *CORL*, 2023. 2
- [35] Jin-Chuan Shi, Miao Wang, Hao-Bin Duan, and Shao-Hua Guan. Language embedded 3d Gaussians for open-vocabulary scene understanding. In *CVPR*, 2024. 1, 2
- [36] Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Buló, Norman Müller, Matthias Nießner, Angela Dai, and Peter Kortschieder. Panoptic lifting for 3d scene understanding with neural fields. In *CVPR*, 2023. 2
- [37] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 2, 6
- [38] Guangming Wang, Lei Pan, Songyou Peng, Shaohui Liu, Chenfeng Xu, Yanzi Miao, Wei Zhan, Masayoshi Tomizuka, Marc Pollefeys, and Hesheng Wang. Nerf in robotics: A survey. *arXiv preprint arXiv:2405.01333*, 2024. 2
- [39] Junjie Wang, Jiemin Fang, Xiaopeng Zhang, Lingxi Xie, and Qi Tian. Gaussianeditor: Editing 3D Gaussians delicately with text instructions. In *CVPR*, 2024. 2
- [40] Yanmin Wu, Jiarui Meng, Haijie Li, Chenming Wu, Yuhao Shi, Xinhua Cheng, Chen Zhao, Haocheng Feng, Er-rui Ding, Jingdong Wang, et al. OpenGaussian: Towards point-level 3D Gaussian-based open vocabulary understanding. *NeurIPS*, 2024. 1, 2, 4, 6, 7
- [41] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. In *Computer Graphics Forum*, 2022. 1
- [42] Teng Xu, Jiamin Chen, Peng Chen, Youjia Zhang, Junqing Yu, and Wei Yang. Tiger: Text-instructed 3D Gaussian retrieval and coherent editing. *arXiv preprint arXiv:2405.14455*, 2024. 2
- [43] Jianglong Ye, Naiyan Wang, and Xiaolong Wang. Feature-erf: Learning generalizable nerfs by distilling foundation models. In *ICCV*, 2023. 1, 2
- [44] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. *ECCV*, 2024. 1, 2, 6, 7
- [45] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *ECCV*, 2024. 2
- [46] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3D Gaussian splatting. In *CVPR*, 2024. 1
- [47] Daiwei Zhang, Gengyan Li, Jiajie Li, Mickaël Bressieux, Otmar Hilliges, Marc Pollefeys, Luc Van Gool, and Xi Wang. EgoGaussian: Dynamic scene understanding from egocentric video with 3D Gaussian splatting. In *3DV*, 2025. 2
- [48] 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. In *CVPR*, 2022. 2
- [49] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. In-place scene labelling and understanding with implicit scene representation. In *ICCV*, 2021. 2
- [50] Chong Zhou, Chen Change Loy, and Bo Dai. Extract free dense labels from CLIP. In *ECCV*, 2022. 2
- [51] 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 *CVPR*, 2024. 1, 2
- [52] 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
- [53] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. EWA volume splatting. In *IEEE VIS*, 2001. 3

## Appendix

The appendix is organized as follows:

- In Sec. A1, we discuss the rationale behind the experiment setup, particularly why we did not report on the ScanNet dataset.
- In Sec. A2, we provide more qualitative results on different datasets.
- In Sec. A3, we provide our implementation details.
- In Sec. A4, we provide more analysis on the necessity of the two-step query.



Figure A1. The original 3D Gaussian Splatting produces low-quality 3D reconstruction on the ScanNet dataset released from OpenGaussian.

### A1. Experimental Design Rationale

For a fair comparison with the concurrent work OpenGaussian, we wanted to have a comparison using the ScanNet dataset as done in their paper. However, we found that their released dataset contains broken camera transformation matrices. Specifically, in the `transforms_train.json` file of `scene0200_00` and `scene0400_00`. Critically, using the provided data, we observed low-quality 3D reconstructions when using 3D Gaussian Splatting, as illustrated in Fig. A1. Therefore, we did not report on ScanNet as the 3D reconstruction setup is already subpar, and the comparison would be focused on the language feature component. Hence, for a third dataset, we chose the Replica dataset to run our experiments which resulted in more accurate 3D reconstruction for titall methods. In this case, the comparison is more meaningful and focused on the language and query components.

### A2. Additional Qualitative Results

We provide additional figures showing the qualitative results. Also attached with this appendix are HTML files showing videos containing additional qualitative comparisons.

**More query results.** Fig. A2, Fig. A3 and Fig. A4 show more object query visualizations from different datasets. As can be seen, our queried objects are more complete and precise across various queries.

**Query results shown in video.** We provide HTML with videos visualizing the queried objects' views from different angles. Please see the file `query_comparison/query_comparison.html`. These visualizations further demonstrate our retrieved objects are more precise and complete than those retrieved by the baseline methods.

**3D point features visualization.** Besides the rendered features shown in the main paper, we now show, Fig. A6, the 3D point features of our method on LERF and 3DOVS datasets. Generally, the features are consistent on the same object, which is beneficial for retrieving objects using our 2-step query pipeline.

**Video visualization of consistent semantic ground truth.** To showcase the ground-truth supervision constructed from our approach, we provide videos illustrating the ground-truth feature  $L_t$  used to supervise the training of language embeddings. Please see `gt_consistency/gt_consistency.html`. We observe that LangSplat's semantic features often struggle with inconsistency when the same object is viewed from different angles. In contrast, our approach produces more consistent and stable semantic features. This consistency benefits training by reducing ambiguity in supervising the same object across varying viewpoints, resulting in more robust and reliable learned language embeddings.

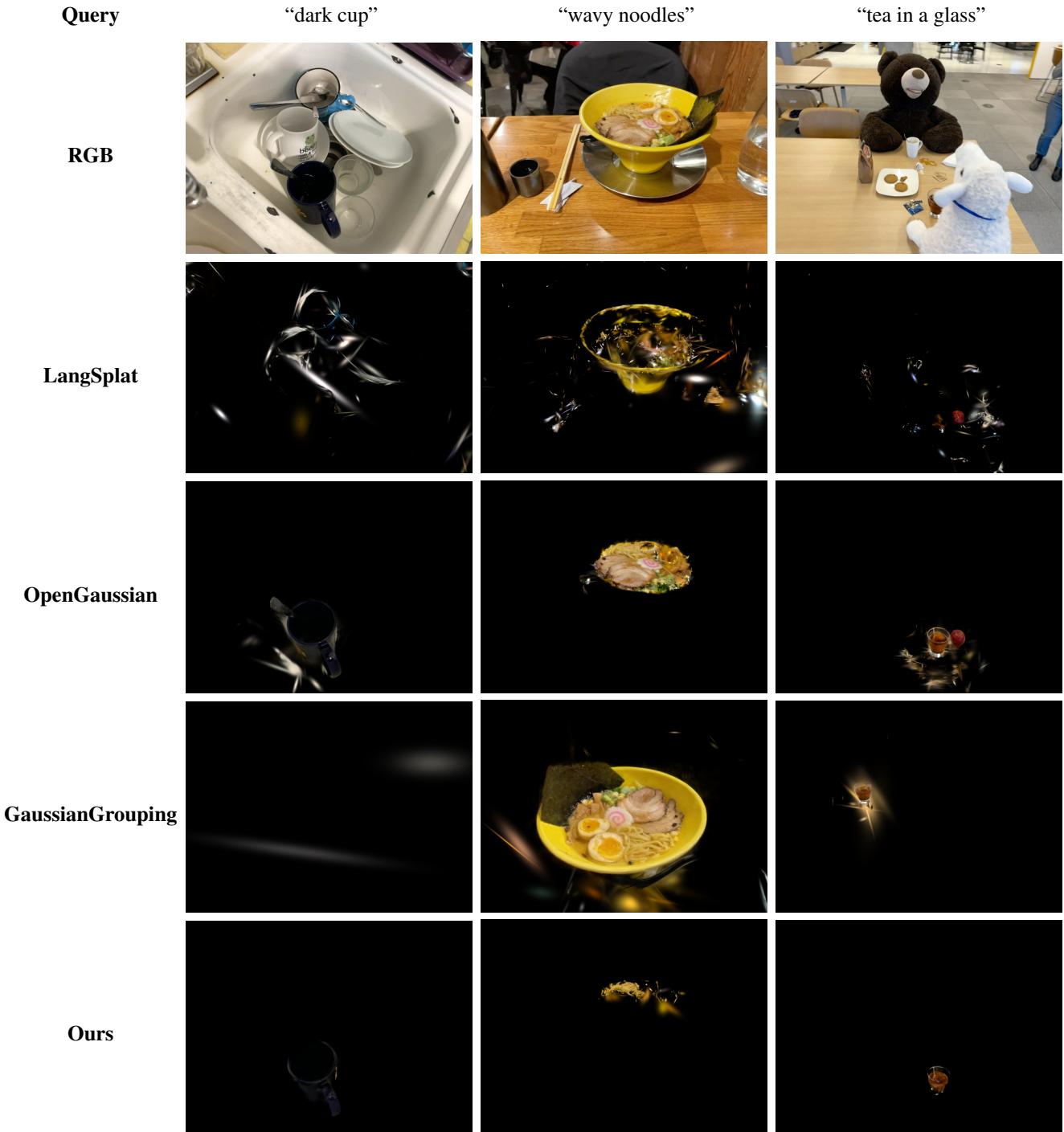


Figure A2. More qualitative results of text query for LERF dataset.

### A3. Implementation Details

**(a) Foundation model specification.** To extract language features, we use the OpenCLIP ViT-B/16 model. For 2D mask segmentation, we employ the SAM ViT-H model, and for tracking masks of the same object, we utilize the SAM2-hiera-large model. During evaluation, our trained model is assessed with SAM level 2 on the LERF dataset and SAM level 3 on the 3D-OVS and Replica datasets.

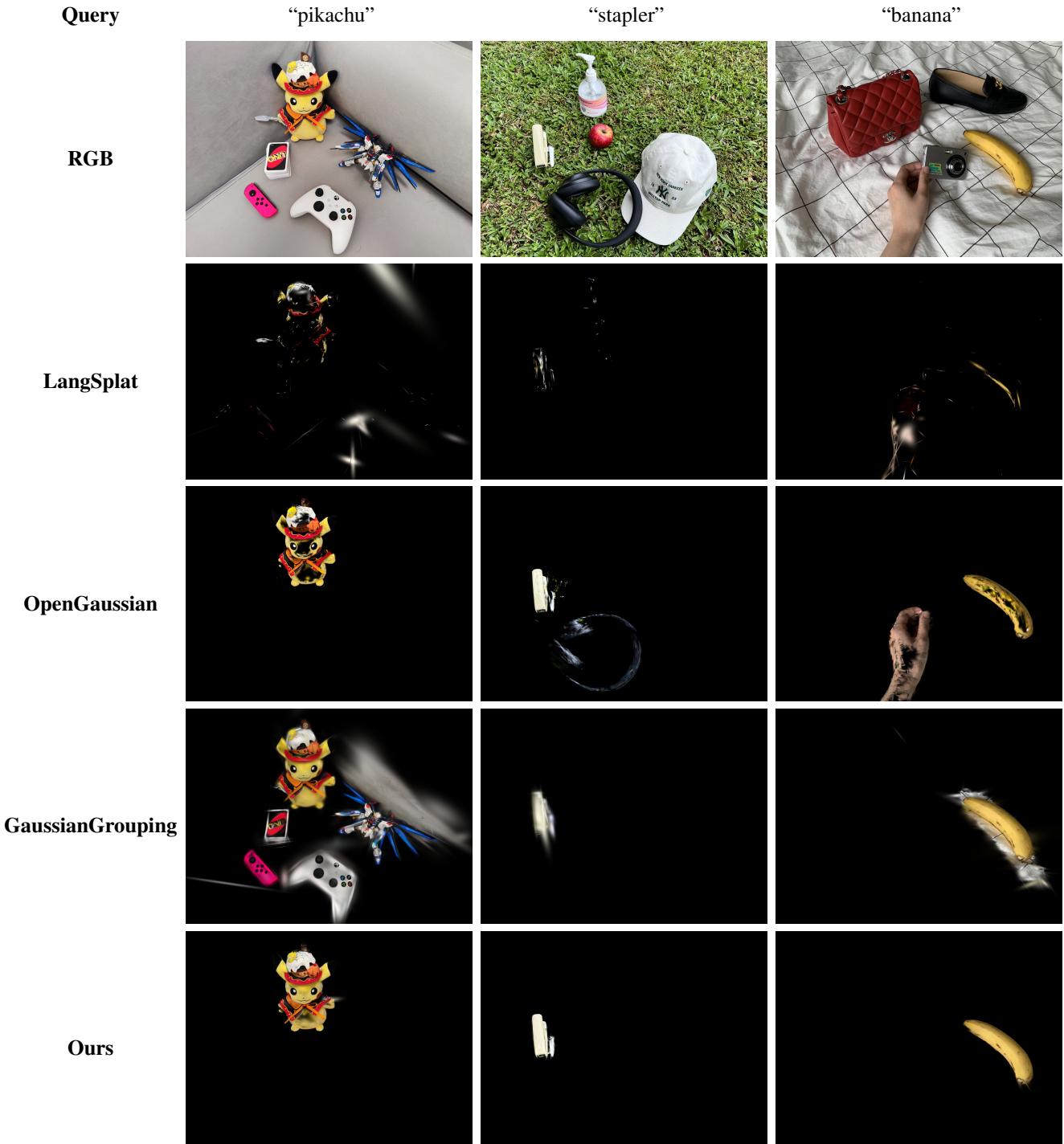


Figure A3. More qualitative results of text query for 3DOVS dataset.

**(b) Training strategy.** For the lightweight autoencoder, we adopt a learning rate of 0.0006 for SAM level 2 and 0.008 for SAM level 3. The structure of the encoder is [256, 128, 128, 64, 32, 3], and decoder is [16, 32, 64, 128, 256, 256, 512]. For 3D scene reconstruction, we follow LangSplat’s approach by pretraining the standard 3D Gaussian Splatting for 30,000 steps. This is followed by training the language embeddings for an additional 30,000 steps, skipping the densification stage. Experiments are conducted on NVIDIA A100 GPU.

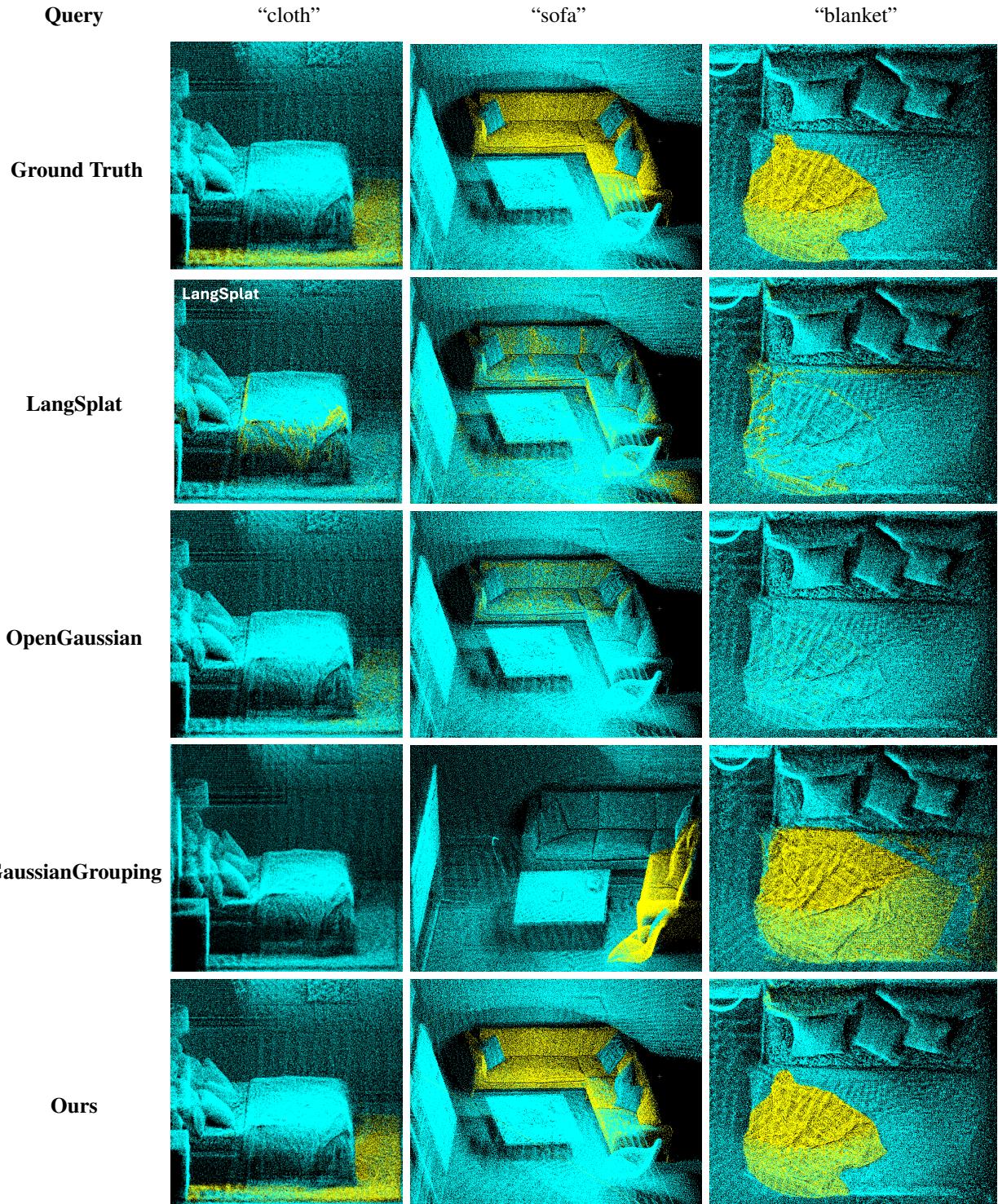


Figure A4. More qualitative results of text query for Replica dataset. Yellow points represent the queried objects.

**(c) Replica dataset.** We select the following scenes from the Replica dataset: `office_0`, `office_1`, `office_2`, `office_3`, `office_4`, `room_0`, `room_1`, `room_2`. To be consistent with OpenGaussian’s evaluation metrics, we skip the densification stage when training the original 3D Gaussian Splatting on the Replica dataset. The 10 categories defined by Replica and used for text queries are: `ceiling`, `sofa`, `cloth`, `stool`, `blanket`, `blinds`, `cushion`,

pot, rug, picture.

**(e) Query thresholding.** During querying, we retain only the points with a cosine similarity to the queried text features exceeding a specified threshold. The threshold is set to 0.995 for LERF, 0.999 for 3DOVS, and 0.95 for the Replica dataset.

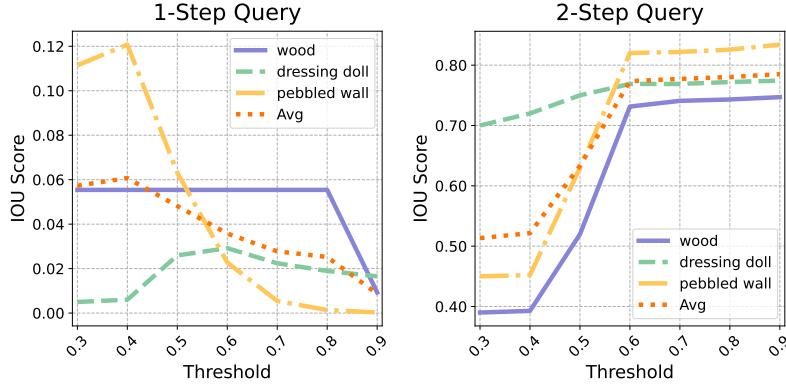


Figure A5. IoU metric per query vs. cosine similarity thresholds. We observe that the 1-step method does not have a suitable threshold for all queries, while our 2-step method does not face this issue.

#### A4. More Analysis on 2-step Query Necessity

In Fig. A5, we visualize the IoU Metric for each of the queries in the scene *bench* in 3D-OVS. We observe that the threshold required for optimal performance varies across different queries. For example, the pebbled wall needs a threshold of 0.4, whereas the dressing doll needs a threshold of 0.6. On average, there is no single threshold that maximizes the performance across all queries. In contrast, our proposed 2-step query does not suffer from this problem. All queries benefit from using a threshold close to 1.

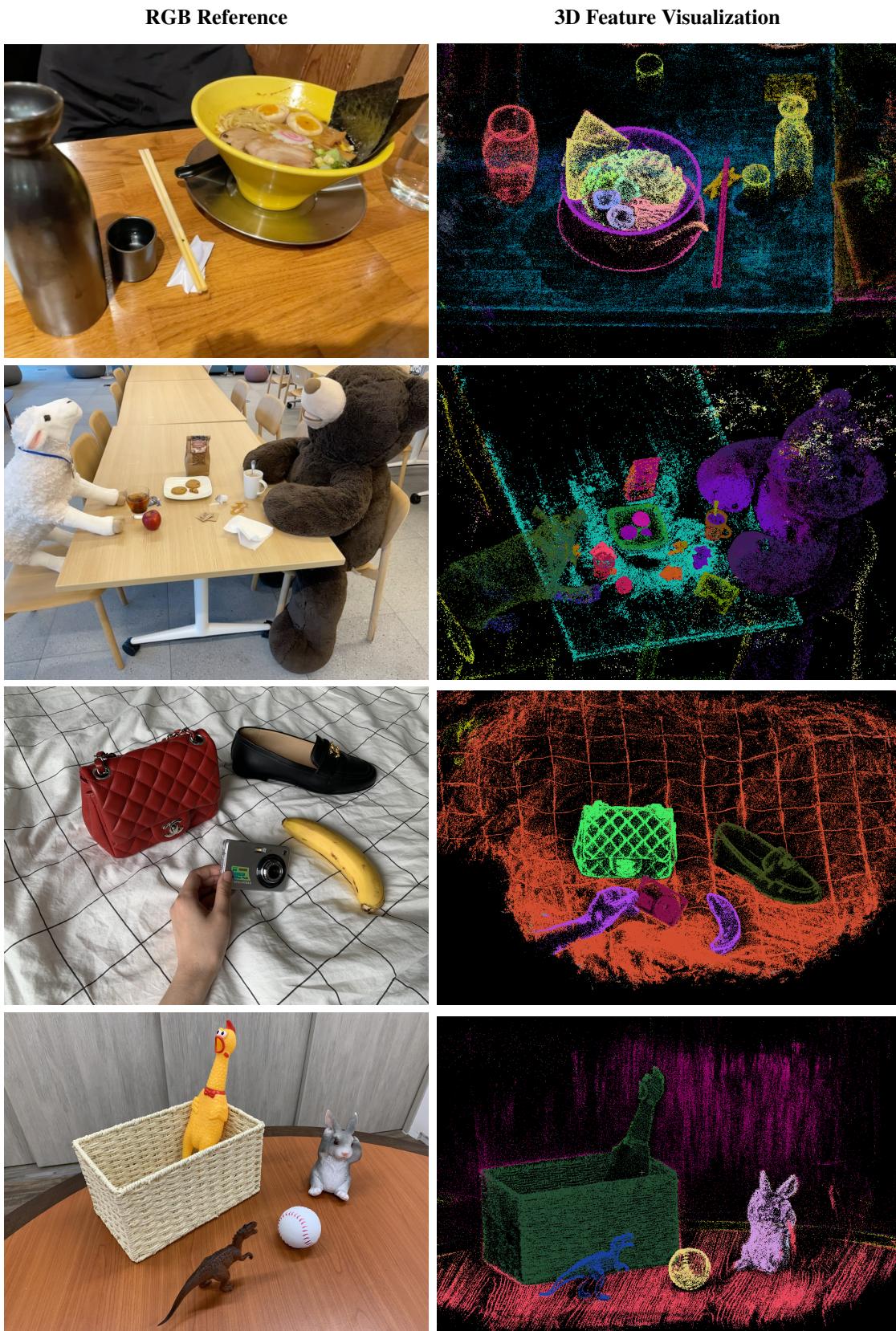


Figure A6. Feature visualization our approach in 3D.