The Thirty-Ninth AAAI Conference on Artificial Intelligence (AAAI-25)

Rethinking Open-Vocabulary Segmentation of Radiance Fields in 3D Space

Hyunjee Lee*, **Youngsik Yun***, **Jeongmin Bae, Seoha Kim, Youngjung Uh**†

Yonsei University

{hyunji12, bbangsik, jaymin.bae, hailey07, yj.uh}@yonsei.ac.kr

Abstract

Understanding the 3D semantics of a scene is a fundamental problem for various scenarios such as embodied agents. While NeRFs and 3DGS excel at novel-view synthesis, previous methods for understanding their semantics have been limited to incomplete 3D understanding: their segmentation results are rendered as 2D masks that do not represent the entire 3D space. To address this limitation, we redefine the problem to segment the 3D volume and propose the following methods for better 3D understanding. We directly supervise the 3D points to train the language embedding field, unlike previous methods that anchor supervision at 2D pixels. We transfer the learned language field to 3DGS, achieving the first real-time rendering speed without sacrificing training time or accuracy. Lastly, we introduce a 3D querying and evaluation protocol for assessing the reconstructed geometry and semantics together. Code, checkpoints, and annotations are available at the project page.

Project page — https://hyunji12.github.io/Open3DRF/

1 Introduction

Semantically understanding 3D space is important for various computer vision tasks. For instance, it is crucial to segment 3D objects accurately for robot manipulation (Rashid et al. 2023; Zheng et al. 2024). Recently, several works have focused on understanding 3D scenes represented by radiance fields such as Neural Radiance Fields (NeRFs) (Mildenhall et al. 2020) and 3D Gaussian Splatting (3DGS) (Kerbl et al. 2023). LERF (Kerr et al. 2023) introduces a language embedding field which is rendered on a chosen viewpoint to be queried by open vocabulary. The language field is supervised by CLIP (Radford et al. 2021) embeddings from the multi-scale patches to capture various sizes of objects. Subsequent works (Zhang, Li, and Ahuja 2024; Qin et al. 2023) incorporate SAM masks (Kirillov et al. 2023) to supervise the language field for clear segmentation boundaries.

We revisit the 3D understanding of NeRFs and 3DGS in four aspects: problem setting, supervision, embeddings, and evaluation. The problem setting of previous works leads to limited 3D understanding as they merely produce rendered 2D masks for given viewpoints rather than 3D semantic volumes. On the other hand, we set the problem to segment 3D volume regarding the semantics of 3D points.

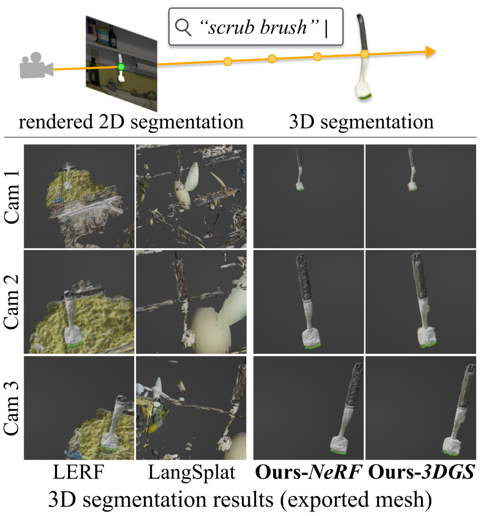

Figure 1: Previous works segment rendered 2D masks on rendered features to understand radiance fields. Instead, we reformulate the task to segment 3D volumes. Our approach significantly improves 3D understanding of radiance fields.

The previous methods encourage the rendered embeddings, rather than the embeddings in 3D space, to match the ground truth. Furthermore, their multi-scale language embeddings require finding the optimal scale for a given viewpoint and a query text, which is prone to multi-view inconsistency. In contrast, we encourage 3D points to learn language embeddings following our revisited problem setting. Although the ground truth is still language embeddings of 2D images, changing the dimension for computing the loss greatly improves 3D understanding as shown in Figure 1. In addition, we remove multi-scale embeddings by encoding masked objects, achieving multi-view consistent rendered feature maps.

Meanwhile, previous works on understanding 3DGS explicitly add language embeddings for each Gaussian and jointly train them (Qin et al. 2023; Zhou et al. 2023). How-

*These authors contributed equally.

†Corresponding author.

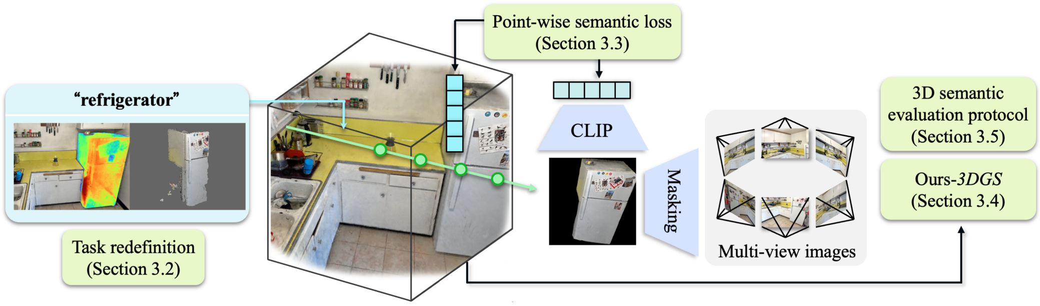

Figure 2: We propose 3D segmentation as a more practical problem setting, segmenting the 3D volume for a given text query (Section 3.2). Then we propose point-wise semantic loss to supervise the sampled point embeddings (Section 3.3) Furthermore, the learned language fields can be transferred into 3DGS for faster rendering speeds (Section 3.4). Lastly, our 3D evaluation protocol measures the 3D segmentation performance both in reconstructed geometry and semantics (Section 3.5).

ever, directly appending a 512-dimensional language embedding to each Gaussian and rendering them result in out-of-memory (OOM) issues. Hence, they either 1) compress language embeddings to low-dimensional features, significantly sacrificing the accuracy (Shi et al. 2023; Qin et al. 2023), or 2) modify the rasterizer, which is slow for training (Zhou et al. 2023). To overcome these drawbacks, we transfer our learned language field into 3DGS to enable the real-time rendering of the accurate language field.

Our final aspect is evaluation. Existing works measure mIoU *in pixels* between the rendered 2D masks and the 2D ground truth, or measure mIoU *on ground truth point clouds* ignoring the reconstructed geometry (Engelmann et al. 2024). However, 3D understanding should know the correct 3D volume of a target semantics. Therefore, the evaluation method should assess semantics and the reconstructed geometry together. Accordingly, we propose to evaluate the accuracy of 3D understanding as the agreement between estimated volume in mesh and ground truth mesh, measured in F1-score. The proposed evaluation is applicable to both NeRF and 3DGS.

In the experiments, we demonstrate the superiority of our method regarding 1) 3D and rendered 2D segmentation accuracy, 2) training and rendering time, and 3) consistency across viewpoints.

In summary, our contributions are:

- We propose a practical problem setting for 3D understanding of NeRFs and 3DGS.
- We propose to directly supervise 3D points before volume rendering to learn a language embedding field. It achieves the state-of-the-art accuracy in 3D and rendered 2D segmentation.
- We propose to transfer the language field to 3DGS. It achieves the first real-time rendering speed among open-vocabulary methods which is 28× faster than the previous fastest method.
- We propose a 3D evaluation protocol between estimated volume and ground truth volume represented as meshes.

2 Related Work

In this section, we briefly review neural 3D scene representations, especially NeRF and 3DGS, and then discuss the semantic understanding of 3D scenes.

Radiance Fields NeRF (Mildenhall et al. 2020) reconstructs a scene as a continuous function that maps 3D points to radiance and density values. Volume rendering pipeline renders the points to determine the color of each pixel on an image from an arbitrary perspective. Rather than volumetric rendering, 3DGS (Kerbl et al. 2023) achieves fast rendering speeds by projecting 3D Gaussians onto the camera plane followed by depth-sorted alpha-blending. Each 3D Gaussian stores location, rotation, scale, opacity, and color. Recent studies use these representations to semantically understand 3D scenes for various applications such as robotics (Rashid et al. 2023; Zheng et al. 2024).

Semantic Understanding of Radiance Field The most straightforward way to understand 3D scenes represented by neural radiance fields is by adding an auxiliary branch for semantic segmentation (Liu et al. 2023; Ye et al. 2023). This approach allows synthesizing semantic masks from novel views but requires a pre-defined list of target classes before training, called *closed*-set. Therefore, retraining or using additional models for untrained queries is necessary, limiting application for open-vocabulary scenarios. As vision-language model (VLM) features (e.g., CLIP (Radford et al. 2021)) expand the semantic understanding to *open*-set, it becomes a widely used approach to distill CLIP features into 3D scenes.

LERF (Kerr et al. 2023) pre-computes multi-scale patches to prepare multi-scale ground-truth CLIP features. Similar to LERF, FMGS (Zuo et al. 2024) pre-computes multi-scale ground-truth CLIP features and averages them to generate a low-resolution hybrid feature map for training. Instead of using patches, LEGaussians (Shi et al. 2023) distills pixel-level CLIP features. Similarly, OpenNeRF (Engelmann et al. 2024) computes pixel-level CLIP features using OpenSeg (Ghiasi et al. 2022). GOI (Qu et al. 2024) trains the codebook to compress the high-dimensional se-

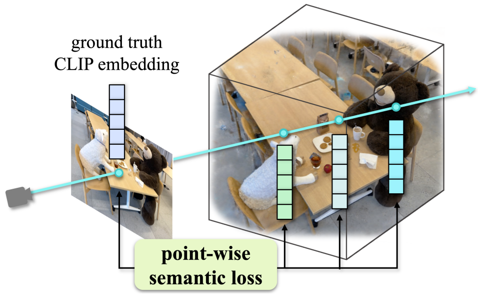

Figure 3: Point-wise semantic loss supervises the language embeddings of sampled points directly in 3D space, ensuring precise semantics.

mantic features into low-dimensional vectors[1]. As it produces mixed CLIP features to contain multiple objects into a patch, LangSplat (Qin et al. 2023) utilizes SAM (Kirillov et al. 2023) to create patches for a single object. OpenMask3D (Takmaz et al. 2023) creates a bounding box from the SAM mask, and crops the image to make per-mask features. However, OpenMask3D uses bounding box, which still leads to mixed CLIP features. Since studies using multi-scale feature maps select the most relevant scale, different scales can be chosen at different views with the same query, leading to multi-view inconsistent results (Kerr et al. 2023; Qin et al. 2023; Shi et al. 2023). On the other hand, we use SAM to make the field free from scales.

Language Embedded 3DGS Directly embedding high-dimensional CLIP features into each Gaussian is infeasible due to the limited GPU shared memory. Therefore, recent studies for understanding open-vocabulary 3D segmentation using 3D as a backbone address this problem in two ways. LEGaussians and LangSplat compress the CLIP features using an autoencoder (Shi et al. 2023; Qin et al. 2023). However, they need to optimize the autoencoder for each scene and endure performance degradation. Others using 3DGS without feature compression (Zhou et al. 2023) utilize global memory, suffering for longer training time. Meanwhile, we effectively optimize 3DGS transferred from our learned language field, without the above trade-off.

3 Methods

First, we provide preliminary on LERF (Kerr et al. 2023). Then we redefine 3D segmentation of radiance fields as computing 3D relevancy scores. Accordingly, we introduce point-wise semantic loss to supervise ray points in 3D space by our scale-free embeddings. In addition, we propose to transfer our learned language field into 3DGS for real-time rendering. Lastly, we introduce the evaluation protocol for 3D segmentation. Figure 2 illustrates an overview.

[1]At the time of this work, the code of GOI was unavailable. However, its released code showed 15 fps on LERF dataset, falling short of real-time rendering.

3.1 Preliminary: LERF

For open-vocabulary segmentation, LERF builds an additional language field on top of iNGP (Müller et al. 2022). The language field F_{lang} is jointly trained with the radiance field by querying point embeddings $F_{\text{lang}}(\mathbf{x}, s)$ at N sample points' position $\mathbf{x}$ and scale s along each ray. The 3D point language embeddings are then accumulated via the volume rendering to obtain rendered language embedding $\hat{\phi}^s_{\text{lang}}$ in 2D pixel space: $\hat{\phi}^s_{\text{lang}} = \sum_{i=0}^{N} w_i F_{\text{lang}}(\mathbf{x}_i, s)$. The weight of each sampled point is calculated as $w_i = T_i(1-\exp(-\sigma_i\delta_i))$, where $T_i = \exp(-\sum_{j=1}^{i-1}\sigma_j\delta_j)$ is transmittance, δ is the distance between adjacent samples, and σ is the volume density. Then, the rendered embedding is normalized to the unit sphere as in CLIP: $\phi^s_{\text{lang}} = \hat{\phi}^s_{\text{lang}}/||\hat{\phi}^s_{\text{lang}}||$ To train the language field, LERF crops the training dataset into multi-scale patches, creating ground truth language embedding $\phi^{\text{gt}_s}_{\text{lang}}$ and maximizing the cosine similarity between the rendered language embedding ϕ_{lang}:

$$L_{\text{lang}} = -\sum_{s} \lambda_{\text{lang}}\, \phi^s_{\text{lang}} \cdot \phi^{\text{gt}_s}_{\text{lang}}. \tag{1}$$

Furthermore, LERF builds an additional branch for the DINO (Caron et al. 2021) feature field as an extra regularizer for achieving clearer object boundaries. Similar to the above, the DINO branch is trained to maximize the similarity between rendered DINO embedding $\hat{\phi}_{\text{dino}}$ and the DINO ground truth $\phi^{\text{gt}}_{\text{dino}}$. The DINO branch is not used during inference.

For rendered 2D segmentation, LERF computes the relevancy score using the rendered embeddings $\hat{\phi}^s_{\text{lang}}$ and obtains the 2D mask:

$$\min_i \frac{\exp(\hat{\phi}^s_{\text{lang}} \cdot \phi_{\text{text}})}{\exp(\hat{\phi}^s_{\text{lang}} \cdot \phi_{\text{text}}) + \exp(\hat{\phi}^s_{\text{lang}} \cdot \phi^i_{\text{canon}})}. \tag{2}$$

3.2 Task Redefinition

Existing methods provide limited 3D semantic understanding for given texts via *rendered* 2D masks on viewpoints that do not directly represent the entire 3D space. Instead, we propose *3D segmentation* as a more practical problem setting: segmenting the 3D volume for given texts. While following the basic elements of language embeddings and relevancy scores, we compute relevancy scores of the language embeddings queried *on 3D points* $\mathbf{x}$ instead of the ones rendered on 2D images:

$$\min_i \frac{\exp(F_{\text{lang}}(\mathbf{x}) \cdot \phi_{\text{text}})}{\exp(F_{\text{lang}}(\mathbf{x}) \cdot \phi_{\text{text}}) + \exp(F_{\text{lang}}(\mathbf{x}) \cdot \phi^i_{\text{canon}})}, \tag{3}$$

where ϕ^i_{canon} represents predefined canonical texts such as `photo` and `image`. In NeRFs, we compute the relevancy scores on the points along the rays through pixels. For 3DGS, we compute the relevancy scores on the center positions of the Gaussians. The regions where the computed relevancy scores surpass the selected threshold are considered object regions.

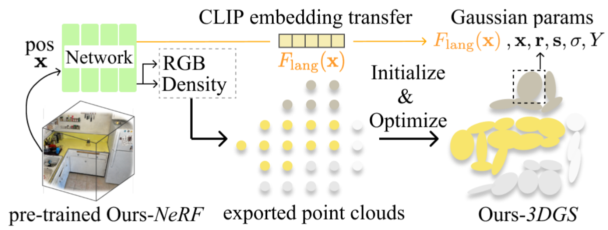

Figure 4: Transferring Ours-*NeRF* into 3DGS: We initialize 3DGS using the point cloud exported from our learned NeRF, then optimize the attributes of 3DGS except for position. The language features obtained by querying the language field at the Gaussian center positions are then transfer to 3DGS.

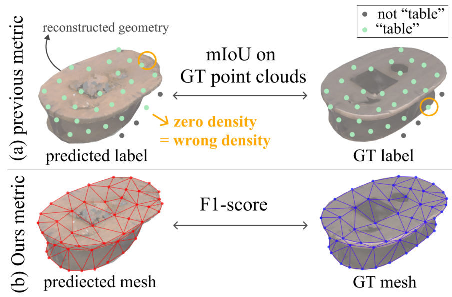

Figure 5: Comparison of 3D Evaluation: (a) Existing methods predict the labels at the ground truth point cloud. It is misleading when the language embeddings capture the object area while the reconstructed geometry does not cover that object area. (b) To address this problem, we extract 3D meshes from the segmented points of the reconstructed scene to measure the F1-score between the exported mesh and the ground truth mesh.

3.3 Supervising Semantics in 3D Space

It is a reasonable choice to minimize the error between the rendered colors and the ground truth colors on 2D images for novel view synthesis. In contrast, the similar objective for language embeddings (Eq. (1)) harms correctly understanding 3D semantics as shown in Figure 1.

To address this issue, we propose a point-wise semantic loss which uses CLIP embeddings $\phi^{\text{gt}}_{\text{lang}}$ as direct ground truth for the embeddings of 3D points on the ray $F_{\text{lang}}(\mathbf{x})$. Specifically, we maximize the similarity between point embeddings $F_{\text{lang}}(\mathbf{x})$ and the ground truth CLIP embedding $\phi^{\text{gt}}_{\text{lang}}$:

$$L_{\text{PS}} = -\sum_{i=0}^{N}(w_i F_{\text{lang}}(\mathbf{x}_i) \cdot \phi^{\text{gt}}_{\text{lang}}). \tag{4}$$

We also apply the same approach described above for DINO regularization. We show that point-wise semantic loss improves not only 3D segmentation but also rendered 2D segmentation thanks to a better understanding of 3D semantics.

Notably, multi-scale approaches (Kerr et al. 2023; Qin et al. 2023) lead to multi-view inconsistency because these approaches have language embedding per scale and different views may have different optimal scales. To address this, we use SAM (Kirillov et al. 2023) to obtain object masks and create scale-free ground truth CLIP embeddings by masking out their background and cropping tightly to the masks. It ensures view-consistent language fields without the need to determine the optimal scale. We note that previous methods with SAM still model multi-scale CLIP embedding fields (Zhang, Li, and Ahuja 2024; Qin et al. 2023).

3.4 Transferring Language Field into 3DGS

A straightforward approach for the same task with 3DGS is to jointly optimize the Gaussians and their additional language embeddings, which requires vast memory consumption. It suffers from slow training[2] (Zhou et al. 2023), or requires an additional model for feature compression, which degrades accuracy due to compression loss (Shi et al. 2023; Qin et al. 2023). Appendix provides more details.

To tackle this problem, we propose to simply bake our learned language field into 3DGS by querying the language embedding at the center of the Gaussians. The queried embeddings are frozen. It runs instantly (22ms) and still allows fast rendering[2].

In addition, we propose to initialize the center coordinates of 3DGS from the learned radiance field. We collect ray points and extract the top 1M points regarding density following NeRFstudio (Tancik et al. 2023). Unlike previous methods (Niemeyer et al. 2024), we implement this approach to align the geometry of 3DGS with the language field. Therefore, we freeze the center position of the Gaussians $\mathbf{x}$, without densification or pruning during training. We then transfer language embeddings $F_{\text{lang}}(\mathbf{x})$ at the center of Gaussians.

Similar to previous methods, we modify the rasterizer to render high-dimensional language features (see Appendix). The rendered language embedding $\hat{\phi}_{\text{lang}}$ is obtained by α-blending:

$$\hat{\phi}_{\text{lang}} = \sum_{i \in N} F_{\text{lang}}(\mathbf{x}_i)\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{5}$$

where N denotes the number of Gaussians overlapping on the pixel. The alpha value is calculated as $\alpha_i = \sigma_i G(\mathbf{x}_i)$, where G denotes the Gaussian kernel and σ denotes the opacity of Gaussians. We use Eq. (3) for 3D segmentation.

3.5 3D Semantic Evaluation Protocol

A previous work (Engelmann et al. 2024) measures the mIoU between the prediction and annotation on the ground truth point cloud. It might incorrectly classify the points with zero density and a correct embedding, shown in Figure 5-(a).

To address this problem, we propose to measure the F1-score between the exported mesh from the segmented results

[2]Training on GPU global memory is slow due to the gradient computation in the backward pass. The forward pass is still fast.

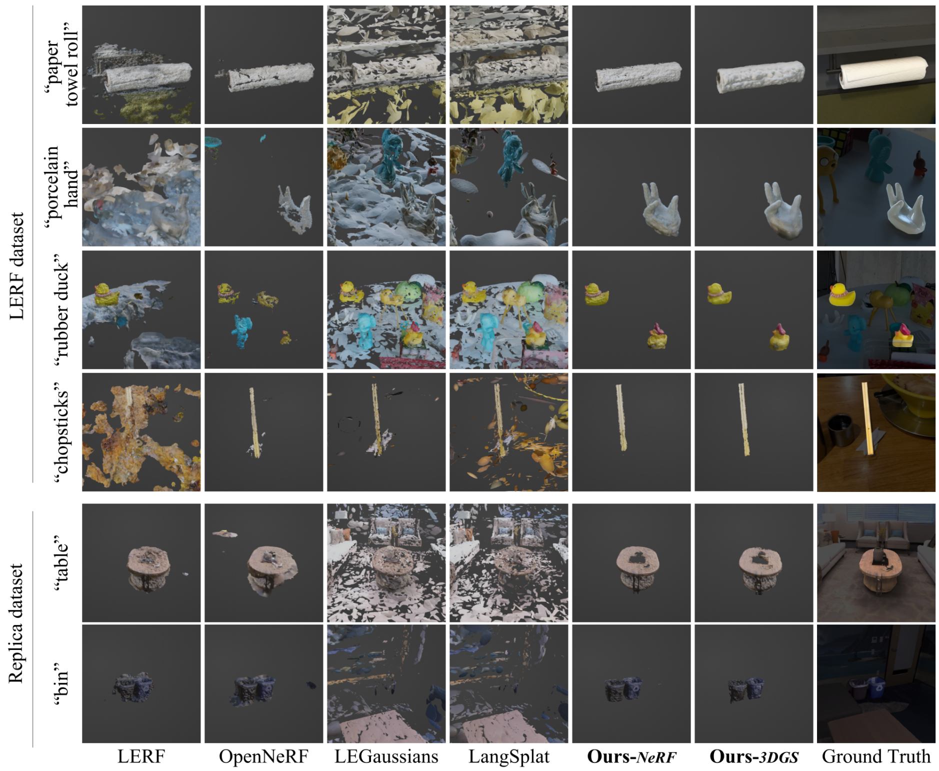

Figure 6: Qualitative comparisons of 3D segmentation on LERF and Replica datasets: We show an exported mesh of 3D querying results for the given text query. Unlike competitors, our method produces more clear boundaries in 3D segmentation results.

to ground truth mesh, inspired by surface reconstruction literature (Knapitsch et al. 2017; Li et al. 2023). Precision computes the ratio of correct volume among the estimated volume. Recall computes the ratio of covered volume among the GT volume. As it is impractical to compute the volume, we approximate the volume by a regular grid points in the estimated and GT meshes. A point is considered *correct* or *covered* if there exists a point within a radius in the counterpart. We note that this protocol can be generally applied to various neural representations, such as NeRF or 3DGS.

Backbone	Method	Replica F1-score ↑
NeRF	LERF	0.0845
	OpenNeRF	0.0361
	Ours-*NeRF*	**0.1520**
3DGS	LEGaussians	0.0067
	LangSplat	0.0087
	Ours-*3DGS*	<u>0.1353</u>

Table 1: 3D Segmentation Accuracy Comparison on the Replica dataset. **Bold** indicates the 1st, and <u>underline</u> indicates the 2nd-best model.

4 Experiments

In this section, we evaluate our methods on various datasets and compare them to competitors in terms of 3D and rendered 2D segmentation with given text queries. We choose LERF, OpenNeRF, LEGaussians, and LangSplat as competitors, which are open-sourced. We re-evalute the competitors using the official code[3]. Appendix provides details of the datasets.

Evaluation For quantitative comparison in 3D segmentation, we export meshes using the Poisson surface reconstruction. In LERF, OpenNeRF, and Ours-*NeRF*, we use Nerfstudio (Tancik et al. 2023) to export meshes with 30K sampled points. In LEGaussians, LangSplat and Ours-*3DGS*, we use SuGaR (Guédon and Lepetit 2023) to export meshes. For rendered 2D segmentation, we evaluate the mIoU and mAP between the ground truth mask and the predicted mask. For qualitative comparison in 3D segmentation, we use 50K points for Nerfstudio in LERF, OpenNeRF, and Ours-*NeRF*.

4.1 Segmentation Accuracy

3D Segmentation We compare qualitative results of 3D segmentation by exporting meshes for the region obtained

[3]The official implementation of LangSplat includes evaluation views during training, while we exclude them for fairness, following LERF and OpenNeRF.

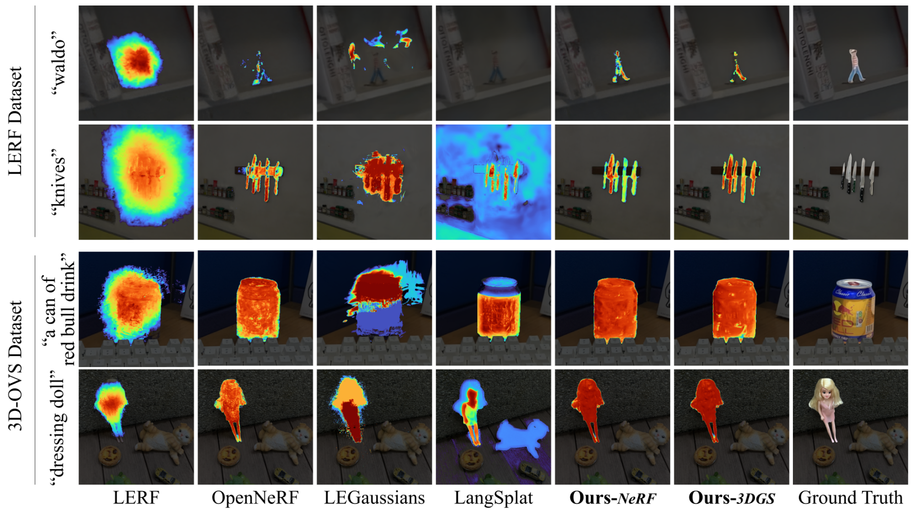

Figure 7: Comparisons of Rendered 2D Segmentation on LERF and 3D-OVS datasets: We show a heatmap of the similarity for the given text query. We dim the background except for the target object, for better visualization. Our method achieves accurate segmentation results compared to competitors.

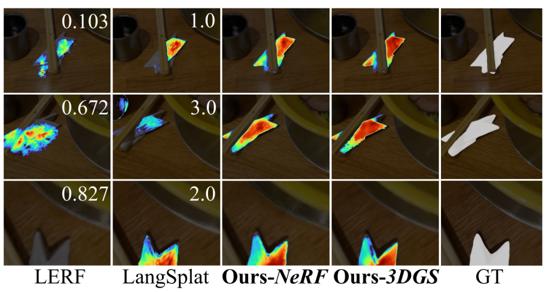

Figure 8: Comparison of view consistency: The figure shows the optimal scales of LERF and LangSplat for each viewpoint on `napkin` in LERF dataset, highlighting significant variation and view inconsistency in competitors. Our method avoids this by utilizing scale-free embeddings.

Method	LERF mIoU ↑	LERF mAP ↑	3D-OVS mIoU ↑	3D-OVS mAP ↑
LERF	31.88	30.44	52.60	57.03
OpenNeRF	26.52	25.84	75.12	75.35
Ours-*NeRF*	**46.37**	**45.86**	<u>77.46</u>	<u>84.65</u>
LEGaussians	21.43	20.81	47.98	50.86
LangSplat[3]	37.53	36.39	74.54	79.49
Ours-*3DGS*	<u>44.37</u>	<u>44.57</u>	**77.51**	**84.86**

Table 2: Quantitative Results of Rendered 2D Segmentation on LERF and 3D-OVS datasets.

through 3D querying (Section 3.2). In Table 1, both of our models surpass the competitors.

In Figure 6, unlike competitors, Ours-*NeRF* and Ours-*3DGS* produce clear segmentation boundaries of the target object from the 3D scene. Notably, both of our models accurately segment complex shape objects and multiple objects like `porcelain hand` and `rubber ducks`. In the `rubber duck` query, LERF, LEGaussians, and LangSplat completely fail to localize the target object, while OpenNeRF incorrectly segments unrelated objects (e.g., a toy elephant and Jake) along with the target object.

Rendered 2D Segmentation In Table 2, both of our models show the highest rendered 2D segmentation performance on LERF and 3D-OVS datasets. Also, we present qualitative results of rendered 2D segmentation by comparing a heatmap for given text queries through 2D querying (Section 3.2). As shown in Figure 7, Ours-*NeRF* and Ours-*3DGS* show clear segmentation boundaries. LangSplat and LEGaussians occasionally fail to find the target object, as seen with `waldo`.

Figure 8 shows relevancy maps and optimal scales along different viewpoints. The float value on the top right of the image represents the optimal scale for LERF and LangSplat from each viewpoint. In Figure 8, our method renders view consistent relevancy maps with `napkin` query. However, LERF and LangSplat show view inconsistent segmentation results due to the changing optimal scale.

4.2 Computational Time

Table 3 shows the computational times of the LERF *waldo_kitchen* scene with a single RTX A5000. We report the training time and rendering time for both ours and the competitors. LangSplat and LEGaussians train an autoencoder for each scene and require decoding during rendering. Note that LangSplat trains 3DGS from scratch, optimizes the language-embedded 3DGS per scale, heavily depends on post-processing (see Appendix). Ours-*3DGS* takes 40 minutes to train Ours-*NeRF*, 5 minutes to optimize 3DGS, and 22 milliseconds to query and transfer language field into

Backbone	Method	Training ↓	Rendering				
			Render ↓	Decode ↓	Post-Process ↓	FPS ↑	Real-Time
NeRF	LERF	40 mins	23688. ms	-	-	0.04	✗
	OpenNeRF	40 mins	5944.7 ms	-	-	0.17	✗
	Ours-*NeRF*	40 mins	2337.7 ms	-	-	0.43	✗
3DGS	LEGaussians	90 mins	15.665 ms	384.6 ms	-	2.50	✗
	LangSplat	100 mins	17.249 ms	0.935 ms	10473 ms	0.10	✗
	Ours-*3DGS*	40+5 mins	14.257 ms	-	-	**70.1**	✓

Table 3: Computational Time: We measure the computational cost on waldo_kitchen scene on an RTX A5000.

Method	Replica	LERF		Computational Cost		
	F1-score ↑	mIoU ↑	mAP ↑	Training Time ↓	FPS ↑	# of Gaussians ↓
Ours-*3DGS*	**0.1354**	**44.37**	**44.57**	**5 mins**	**70.1**	**1M**
w/o NeRF init	0.1108	36.14	36.74	13 mins	41.2	2M

Table 4: Ablation Study of Initializing NeRF into 3DGS: The ablation is conducted on Replica dataset for 3D segmentation and on LERF dataset for rendered 2D segmentation. We measure the computational cost on *waldo_kitchen* scene with RTX A5000.

Method	Replica	LERF	
	F1-score ↑	mIoU ↑	mAP ↑
Ours full	**0.1532**	**46.37**	**45.86**
w/o L_{PS}	0.0537	44.50	44.05
w/o $F_{\text{lang}}(\mathbf{x}_i)$	0.1114	30.03	28.85

Table 5: Ablation Study of Supervising Semantics in 3D Space: The datasets are Replica and LERF for 3D and rendered 2D segmentation, respectively. w/o L_{PS} denotes using LERF CLIP loss L_{lang} instead of point-wise semantic loss and w/o $F_{\text{lang}}(\mathbf{x}_i)$ denotes using multi-scale embedding $F_{\text{lang}}(\mathbf{x}_i, s)$ instead of scale-free embedding.

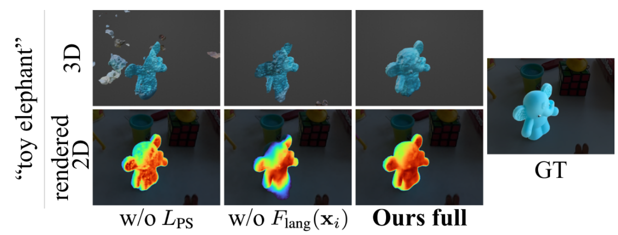

Figure 9: Qualitative Results of Ablation Study on *figurines* scene in LERF dataset.

3DGS. Ours-*3DGS* achieves real-time rendering of language embeddings for the first time, 28× faster than LEGaussians.

4.3 Ablation Study

Point-wise Semantic Loss We demonstrate the necessity of our point-wise semantic loss. Figure 9 qualitatively shows improvements due to point-wise semantic loss compared to 2D supervision with Eq. (1). Point-wise semantic loss removes meaningless floaters related to depth ambiguity. Table 5 shows that leveraging point-wise semantic loss instead of Eq. (1) makes better understanding in 3D space, improving the F1-score (+0.0995). Furthermore, point-wise semantic loss also indirectly improves rendered 2D segmentation accuracy.

Scale-Free Embedding Our scale-free language embedding field from SAM-segmented objects produce greatly improved accuracy with clear object boundaries compared to multi-scale embedding field. Figure 9 shows that scale-free embedding helps covering the head of the toy elephant which was lost with multi-scale embedding. In Table 5, the quantitative performance of the 3D and rendered 2D segmentation greatly improves using scale-free embedding.

NeRF Initialization We demonstrate the necessity of initializing 3DGS with the extracted point clouds from our learned NeRF. As shown in Table 4, training 3DGS with SfM (Schönberger and Frahm 2016) initialization with the densification and pruning leads to performance degradation in both 3D and rendered 2D segmentation. This indicates that aligning geometry and semantics is essential for transferring the language field to 3DGS. Moreover, NeRF initialization produces fewer Gaussians (0.50×) which lead to less training time (0.38×) and faster rendering speed (1.70×).

5 Conclusion

We revisit the current literature on 3D understanding of NeRF and 3DGS and revise the problem setting. We reformulate the task to produce 3D segmented volumes instead of rendered 2D masks and propose a 3D evaluation protocol. We achieve state-of-the-art segmentation accuracy in both 3D and rendered 2D by computing loss directly on 3D points. Moreover, we enable the first real-time rendering speed among open-vocabulary methods by transferring the learned language field to 3DGS. We hope this paper drives forward a better 3D understanding of radiance fields by reconsidering the problem set.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. RS-2024-00439762, Developing Techniques for Analyzing and Assessing Vulnerabilities, and Tools for Confidentiality Evaluation in Generative AI Models)

References

Caron, M.; Touvron, H.; Misra, I.; Jégou, H.; Mairal, J.; Bojanowski, P.; and Joulin, A. 2021. Emerging Properties in Self-Supervised Vision Transformers. arXiv:2104.14294.

Engelmann, F.; Manhardt, F.; Niemeyer, M.; Tateno, K.; and Tombari, F. 2024. OpenNerf: Open Set 3D Neural Scene Segmentation with Pixel-Wise Features and Rendered Novel Views. In *The Twelfth International Conference on Learning Representations*.

Ghiasi, G.; Gu, X.; Cui, Y.; and Lin, T.-Y. 2022. Scaling Open-Vocabulary Image Segmentation with Image-Level Labels. arXiv:2112.12143.

Guédon, A.; and Lepetit, V. 2023. SuGaR: Surface-Aligned Gaussian Splatting for Efficient 3D Mesh Reconstruction and High-Quality Mesh Rendering. *arXiv preprint arXiv:2311.12775*.

Kerbl, B.; Kopanas, G.; Leimkühler, T.; and Drettakis, G. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4).

Kerr, J.; Kim, C. M.; Goldberg, K.; Kanazawa, A.; and Tancik, M. 2023. LERF: Language Embedded Radiance Fields. In *International Conference on Computer Vision (ICCV)*.

Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A. C.; Lo, W.-Y.; Dollár, P.; and Girshick, R. 2023. Segment Anything. *arXiv:2304.02643*.

Knapitsch, A.; Park, J.; Zhou, Q.-Y.; and Koltun, V. 2017. Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction. *ACM Transactions on Graphics*, 36(4).

Li, Z.; Müller, T.; Evans, A.; Taylor, R. H.; Unberath, M.; Liu, M.-Y.; and Lin, C.-H. 2023. Neuralangelo: High-Fidelity Neural Surface Reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, K.; Zhan, F.; Zhang, J.; Xu, M.; Yu, Y.; Saddik, A. E.; Theobalt, C.; Xing, E.; and Lu, S. 2023. Weakly Supervised 3D Open-vocabulary Segmentation. *arXiv preprint arXiv:2305.14093*.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*.

Müller, T.; Evans, A.; Schied, C.; and Keller, A. 2022. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.*, 41(4): 102:1–102:15.

Niemeyer, M.; Manhardt, F.; Rakotosaona, M.-J.; Oechsle, M.; Duckworth, D.; Gosula, R.; Tateno, K.; Bates, J.; Kaeser, D.; and Tombari, F. 2024. RadSplat: Radiance Field-Informed Gaussian Splatting for Robust Real-Time Rendering with 900+ FPS. *arXiv.org*.

Qin, M.; Li, W.; Zhou, J.; Wang, H.; and Pfister, H. 2023. LangSplat: 3D Language Gaussian Splatting. *arXiv preprint arXiv:2312.16084*.

Qu, Y.; Dai, S.; Li, X.; Lin, J.; Cao, L.; Zhang, S.; and Ji, R. 2024. GOI: Find 3D Gaussians of Interest with an Optimizable Open-vocabulary Semantic-space Hyperplane. arXiv:2405.17596.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; Krueger, G.; and Sutskever, I. 2021. Learning Transferable Visual Models From Natural Language Supervision. arXiv:2103.00020.

Rashid, A.; Sharma, S.; Kim, C. M.; Kerr, J.; Chen, L. Y.; Kanazawa, A.; and Goldberg, K. 2023. Language Embedded Radiance Fields for Zero-Shot Task-Oriented Grasping. In *7th Annual Conference on Robot Learning*.

Schönberger, J. L.; and Frahm, J.-M. 2016. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Shi, J.-C.; Wang, M.; Duan, H.-B.; and Guan, S.-H. 2023. Language Embedded 3D Gaussians for Open-Vocabulary Scene Understanding. arXiv:2311.18482.

Takmaz, A.; Fedele, E.; Sumner, R. W.; Pollefeys, M.; Tombari, F.; and Engelmann, F. 2023. OpenMask3D: Open-Vocabulary 3D Instance Segmentation. arXiv:2306.13631.

Tancik, M.; Weber, E.; Ng, E.; Li, R.; Yi, B.; Kerr, J.; Wang, T.; Kristoffersen, A.; Austin, J.; Salahi, K.; Ahuja, A.; McAllister, D.; and Kanazawa, A. 2023. Nerfstudio: A Modular Framework for Neural Radiance Field Development. In *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH '23.

Ye, M.; Danelljan, M.; Yu, F.; and Ke, L. 2023. Gaussian Grouping: Segment and Edit Anything in 3D Scenes. *arXiv preprint arXiv:2312.00732*.

Zhang, H.; Li, F.; and Ahuja, N. 2024. Open-NeRF: Towards Open Vocabulary NeRF Decomposition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 3456–3465.

Zheng, Y.; Chen, X.; Zheng, Y.; Gu, S.; Yang, R.; Jin, B.; Li, P.; Zhong, C.; Wang, Z.; Liu, L.; Yang, C.; Wang, D.; Chen, Z.; Long, X.; and Wang, M. 2024. GaussianGrasper: 3D Language Gaussian Splatting for Open-vocabulary Robotic Grasping. arXiv:2403.09637.

Zhou, S.; Chang, H.; Jiang, S.; Fan, Z.; Zhu, Z.; Xu, D.; Chari, P.; You, S.; Wang, Z.; and Kadambi, A. 2023. Feature 3DGS: Supercharging 3D Gaussian Splatting to Enable Distilled Feature Fields. *arXiv preprint arXiv:2312.03203*.

Zuo, X.; Samangouei, P.; Zhou, Y.; Di, Y.; and Li, M. 2024. FMGS: Foundation Model Embedded 3D Gaussian Splatting for Holistic 3D Scene Understanding. arXiv:2401.01970.