

# OpenGS-SLAM: Open-Set Dense Semantic SLAM with 3D Gaussian Splatting for Object-Level Scene Understanding

Dianyi Yang<sup>1,2</sup>, Yu Gao<sup>1,2</sup>, Xihan Wang<sup>1,2</sup>, Yufeng Yue<sup>1,2</sup>, Yi Yang<sup>\*,1,2</sup>, Mengyin Fu<sup>1,2</sup>

**Abstract**—Recent advancements in 3D Gaussian Splatting have significantly improved the efficiency and quality of dense semantic SLAM. However, previous methods are generally constrained by limited-category pre-trained classifiers and implicit semantic representation, which hinder their performance in open-set scenarios and restrict 3D object-level scene understanding. To address these issues, we propose OpenGS-SLAM, an innovative framework that utilizes 3D Gaussian representation to perform dense semantic SLAM in open-set environments. Our system integrates explicit semantic labels derived from 2D foundational models into the 3D Gaussian framework, facilitating robust 3D object-level scene understanding. We introduce Gaussian Voting Splatting to enable fast 2D label map rendering and scene updating. Additionally, we propose a Confidence-based 2D Label Consensus method to ensure consistent labeling across multiple views. Furthermore, we employ a Segmentation Counter Pruning strategy to improve the accuracy of semantic scene representation. Extensive experiments on both synthetic and real-world datasets demonstrate the effectiveness of our method in scene understanding, tracking, and mapping, achieving 10× faster semantic rendering and 2× lower storage costs compared to existing methods. Project page: <https://young-bit.github.io/opengs-github.github.io/>.

## I. INTRODUCTION

Dense semantic Simultaneous Localization and Mapping (SLAM) is a fundamental challenge in robotics [1], [2] and embodied intelligence [3], [4]. It integrates semantic understanding of the environment into 3D scene reconstruction and estimates camera pose simultaneously. Traditional semantic SLAM struggles with predicting unknown areas and requires significant map storage [5]. While NeRF-based methods [6]–[9] mitigated these issues, they still suffer from inefficient per-pixel raycasting rendering [10].

Alternative to NeRFs, the recently emerged 3D Gaussian Splatting (3DGS) [11] has shown remarkable reconstruction quality with high training and rendering efficiency. Following these advantages, 3DGS-based Semantic SLAM methods [12], [13] are developed to achieve high-quality semantic mapping. These methods embed 2D semantic features into 3DGS representation, where each Gaussian is enhanced with an N-channel feature embedding.

However, these methods often rely on pre-trained classifiers with a limited number of categories to produce 2D segmentation results, which limits their effectiveness in open-set scenarios. Further, since all semantic information is

\*This work was partly supported by National Key R&D Program of China (2022YFC2603600) and National Natural Science Foundation of China (Grant No. NSFC 62233002)

<sup>1</sup>School of Automation, Beijing Institute of Technology, Beijing, China

<sup>2</sup>National Key Lab of Autonomous Intelligent Unmanned Systems, Beijing Institute of Technology, Beijing, China

\*Corresponding author: Y. Yang Email: yang\_yi@bit.edu.cn

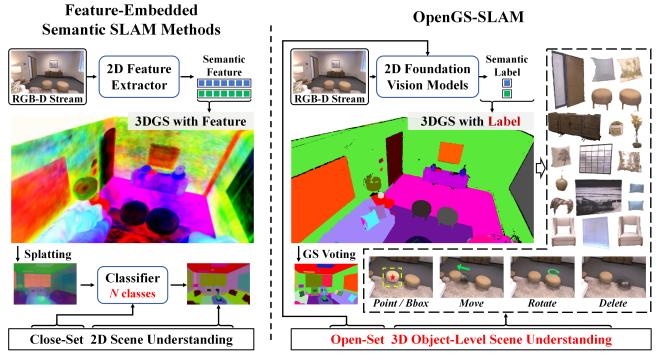


Fig. 1: Compared to the feature-embedded methods [12], [13], our approach integrates semantic labels into the 3D Gaussian scene representation, ensuring that Gaussians belonging to the same object are consistently labeled. This enables more effective 3D object-level scene understanding and interaction. By leveraging 2D foundational vision models, our approach facilitates open-set dense semantic SLAM. The images on the left are from [12].

implicitly stored within the feature-embedded representation, it becomes challenging to directly access the semantic label of each Gaussian. This makes them incompatible with embodied intelligence that necessitates 3D object-level scene understanding and interaction.

To address these limitations and extend semantic SLAM to open-set scenarios, we propose OpenGS-SLAM, a novel framework that incorporates 2D foundational vision models and assigns an explicit semantic label to each Gaussian, while enabling online semantic mapping. The foundational model in this framework is easily customizable; for example, it can be replaced with SAM [14] or MobileSAMv2 [15] to ensure flexibility. Additionally, the framework can map semantic labels generated by these models into 3D Gaussians, facilitating its application in open-set scenarios. By assigning an explicit semantic label to each Gaussian, the framework efficiently supports 3D object-level scene understanding.

During the development of this framework, we faced three major challenges: **1)** the non-differentiable nature of the semantic label attribute prevents us from using 3DGS rasterization for fast label map rendering and scene updating; **2)** inconsistencies in SAM’s results across multiple views cause the same object to receive varying labels or be segmented into parts or wholes; and **3)** during training, Gaussians in regions with less view constraint may expand significantly, resulting in inaccurate segmentation.

Focusing on these challenges, **1)** we introduce Gaussian

Voting Splatting (GS Voting) to facilitate rapid 2D label map rendering and scene updating. **2)** To achieve consistent segmentation and efficiently incorporate new semantic information into the scene, we propose a Confidence-based 2D Label Consensus method. **3)** We propose a Segmentation Counter Pruning strategy to achieve more accurate segmentation of the scene. Additionally, we designed an ensemble semantic information generator to produce 2D semantic labels.

To summarize, our contributions are as follows:

- We propose OpenGS-SLAM, which enables open-set semantic SLAM based on 3D Gaussian Splatting. It supports a wide range of off-the-shelf 2D foundational vision models without extra training.
- We propose a novel semantic 3DGS representation with the Gaussian Voting Splatting method, which enables 3D object-level scene understanding with **10x faster** rendering from novel views, **2x lower** storage costs and more accurate semantic segmentation compared to other methods.
- Our method achieves competitive performance in both synthetic and real-world scenarios in terms of tracking, mapping, and scene understanding.

## II. RELATED WORK

**Gaussian Splatting based RGB-D SLAM.** 3D Gaussian Splatting [11] revolutionized SLAM by enabling high-fidelity scene reconstruction and precise pose estimation. Notable advances include SplaTAM [10] with silhouette-guided optimization, GS-SLAM’s [16] adaptive Gaussian expansion, and GS-ICP SLAM’s G-ICP for accurate pose tracking. Additionally, 3DGS has been extended for semantic scene understanding, with methods like SGS-SLAM [17] incorporating 2D semantic priors. SemGauss-SLAM [12] and NEDS-SLAM [13] further optimize memory usage through embedding low-dimensional semantic vectors into the 3D Gaussian framework. However, the reliance on auxiliary classifiers or lightweight encoders for 2D segmentation limits their performance in open-set scenarios.

**2D Foundation Vision Models.** These methods are designed to capture broad knowledge from large-scale data and perform zero-shot tasks on unseen data. The Recognize Anything Model [18] can generate open-set classes from images. Segment Anything [14] excels in zero-shot segmentation, with advancements like MobileSAMv2 [15] achieving over 20 fps in full-scene segmentation. SAM2.0 [19] extends to real-time video segmentation but struggles with new object detection and multi-object tracking. Methods like Yolo-world [20] and Grounding Dino [21] enable zero-shot object detection. Some approaches [22], [23] integrate these models for zero-shot semantic segmentation.

**Open-World 3DGS Scene Understanding Method.** Recent methods in this field focus on lifting information from 2D foundation models to build 3D representations in an offline manner. For instance, LangSplat [24], Language-Embedded [25], and Feature-3DGS [26] leverage CLIP [27] and SAM [14] for 2D pixel-level understanding. In contrast, GS-Grouping [28] and OpenGaussian [29] elevate SAM’s 2D segmentation for 3D segmentation, with GS-Grouping

using DEVA [30] to resolve frame-to-frame inconsistencies. However, these methods rely on time-consuming processes, limiting their applicability for online reconstruction.

## III. METHOD

### A. Ensemble Semantic Information Generator

To fully exploit the capabilities of existing 2D foundational models in open-set scenarios, we integrate three types of expert models designed for specific contexts. The input RGB images are first fed into a Recognize Anything Model (RAM) [18] and a SAM model. The RAM generates open-set classes information, which serves as a text prompt for the Detection Model (using YOLO-World [20]) to produce semantic bounding boxes and detection scores. The SAM model outputs a label map with confidence scores, though the labels are randomly assigned. For each label, we match the bounding box with the highest IoU (above 0.5) to establish semantic correspondence, which is recorded in an input Label-Class Table  $\mathcal{M}_{LC}^I$ . We also maintain a global Label-Class Table  $\mathcal{M}_{LC}^G$  throughout the SLAM process. For labels without a matching semantic bounding box, we assign them to the *None* class with a detection score of 0.0. The expert models are customizable; for instance, we test SAM1.0 [14] and MobileSAMv2 [15] in the experimental section to assess their performance and adaptability in various scenarios. Notably, the categories in these tables are not predefined; they begin from an empty set and can expand dynamically during the SLAM process.

### B. Semantic Gaussian Mapping

In this section, we first introduce our scene representation strategy. Next, we explain how to render both RGB-D images and label maps using Gaussian Voting Splatting. Finally, we describe the process of scene densification and updating.

**3D Gaussian strategy.** We employ a set of Gaussians with novel designed properties for scene representation. Compared with all previous methods, we replaced the feature embeddings in each Gaussian with a one-dimensional, non-differentiable attribute named *GS Label*. Specifically, each Gaussian includes a 3D center position  $x$ , covariance  $\Sigma$ , opacity  $o$ , color  $f$ , and label  $\ell$ .

**Gaussian Voting Splatting.** This method combines the strengths of 3D Gaussian Splatting and our semantic 3DGS scene representation to achieve high-fidelity RGB-D rendering alongside precise semantic mapping. First, with the optimized 3D Gaussian scene representation, each Gaussian is projected from 3D space onto the 2D image plane using the camera pose  $T = \{R, t\}$ . The covariance matrix  $\Sigma$  is transformed to the 2D plane as:

$$\Sigma' = JT^{-1}\Sigma T^{-T}J^T, \quad (1)$$

where  $J$  represents the Jacobian of the projection function. Second, the Gaussians are sorted by depth and rendered using front-to-back alpha compositing. The pixel color  $C(p)$  and depth  $D(p)$  are calculated through  $\alpha$ -blending, as denoted in Eq.2, where  $f_i$  and  $d_i$  denote the color and depth of the

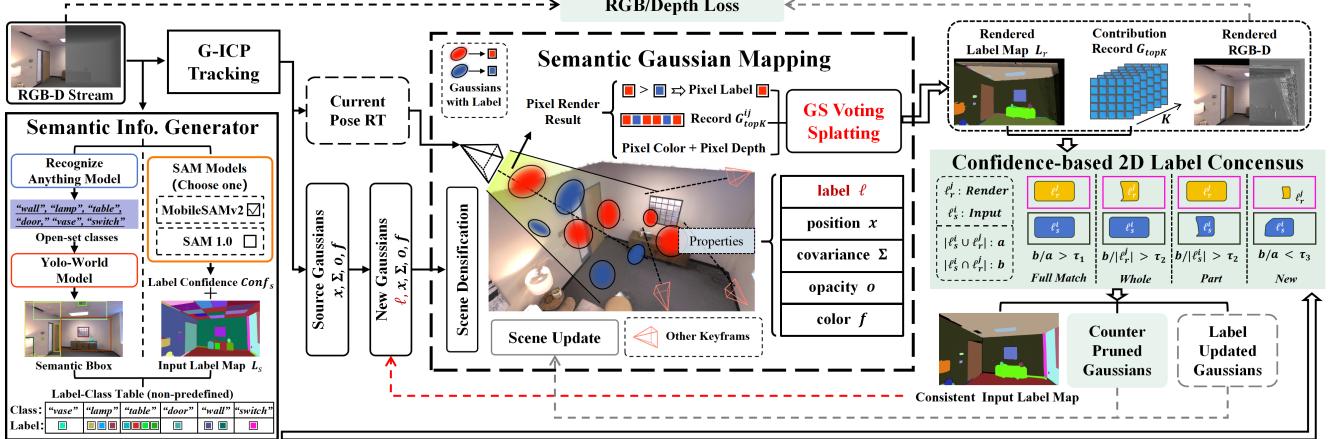


Fig. 2: **An overview of OpenGS-SLAM.** Our method takes an RGB-D stream as input. RGB images are first processed by the Semantic Information Generator and G-ICP to extract semantic information and estimate the current pose. Using this pose, we perform precise and efficient semantic rendering via Gaussian Voting Splatting. We then unify the input label map with the current map through Confidence-based 2D Label Consensus, ensuring semantic consistency. During this process, partial Gaussian data is updated, and counter Gaussians are pruned.

$i$ -th Gaussian, respectively, and  $\alpha_i$  is the opacity-weighted density, derived from the 2D covariance  $\Sigma'$  and opacity  $o_i$ .

$$\begin{cases} C(p) = \sum_{i \in N} f_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \\ D(p) = \sum_{i \in N} d_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j) \end{cases} \quad (2)$$

Third, for each pixel  $p$  in the final rendered label map, we aggregate all contributing Gaussians, denoted as  $G_p$ , and apply label-aware  $\alpha$ -blending to determine the contribution of each label. Specifically, let  $w_j^i$  represent the weight of the  $i$ -th Gaussian  $g_i$  with label  $\ell_j$  in  $G_p$ , computed as:

$$w_j^i = \alpha_i \prod_{k=1}^{i-1} (1 - \alpha_k) \quad (3)$$

where  $k$  refers to the depth-sorted index of Gaussians in  $G_p$ . Defining  $G_p^j$  as the set of Gaussians in  $G_p$  with  $\ell_j$ , the total weight  $W_j$  for  $\ell_j$  is computed as:

$$W_j = \sum_{g_i \in G_p^j} w_j^i. \quad (4)$$

The final label assigned to pixel  $p$  corresponds to the label with the highest cumulative weight. This approach ensures that each pixel is labeled based on the most significant cumulative contribution from the Gaussians, providing accurate and efficient 2D label map rendering.

**Scene Densification and Updating.** For densification, once a frame is identified as a keyframe, we use the covariance computed from G-ICP to initialize the source Gaussians [16]. After integrating the semantic *GS Label* attribute, these Gaussians are added into the map.

For updating, we optimize the differentiable parameters using a loss function applied to selected frames following

[16]. In contrast, the non-differentiable *GS Label* is updated in an explicit manner. During the label map rendering, we record the top  $K = 50$  Gaussians contributing to each pixel, generating the matrix  $G_{topK} \in \mathbb{R}^{H \times W \times K}$ . If a pixel  $p$  labeled as  $\ell_s$  in the rendered label map, is updated to  $\ell_t$ , we utilize  $G_{topK}$  to identify the Gaussians previously labeled as  $\ell_s$  that contribute to pixel  $p$  and update their label to  $\ell_t$ . This approach ensures efficient and precise updates of semantic labels across the scene.

### C. Confidence-based 2D Label Consensus

Currently, models like MobileSAMv2 [15] that support real-time full-scene segmentation struggle with multi-view inconsistencies, where the same object can end up with different labels or be segmented into parts/whole across different viewpoints. Most existing methods address this through additional tracking models [28] or global maps [26], [31], limiting their real-time applicability. Therefore, we propose a Confidence-based 2D Label Consensus method, leveraging the efficient 3DGS scene rendering to unify input semantic information with the current map on-the-fly.

We define  $\mathcal{L}_s = \{\ell_s^i\}_{i=1}^M$  as the set of labels in the input label map, with  $\mathcal{C}_s$  representing the confidence of each label. Similarly,  $\mathcal{L}_r = \{\ell_r^j\}_{j=1}^N$  denotes the set of labels in the rendered label map from the current viewpoint, with  $\mathcal{C}_r$  as the corresponding confidence. Additionally, we define  $|\ell|$  as the number of pixels associated with label  $\ell$  in its respective label map. Our goal is to achieve label consensus by matching all labels in  $\mathcal{L}_s$  with those in  $\mathcal{L}_r$ , or creating new labels if necessary. The final output is a Consistent Input Label Map that provides consistent semantic information.

For any  $\ell_s^i \in \mathcal{L}_s$  and  $\ell_r^j \in \mathcal{L}_r$ , we identify four possible relationships based on their  $|\cdot|$  in the label map, as shown in Fig.2, where we set three threshold as  $\tau_1, \tau_2, \tau_3$ .

- *Full Match*: If  $\ell_s^i$  fully matches  $\ell_r^j$ , we update the label in  $\mathcal{L}_s$  from  $\ell_s^i$  to  $\ell_r^j$  and set  $\mathcal{C}_r^j$  to the maximum of its current

value and  $\mathcal{C}_s^i$ .

- **Partial Match:** We define  $\mathcal{P}_s^j \in \mathcal{L}_s$  as all part labels that partially matches to  $\ell_r^j$ . We then calculate their area-weighted average confidence  $\bar{\mathcal{C}}_s$ , computed as:

$$\bar{\mathcal{C}}_s = \frac{\sum_{\ell_s^i \in \mathcal{P}_s^j} \mathcal{C}_s^i \times |\ell_s^i|}{\sum_{\ell_s^i \in \mathcal{P}_s^j} |\ell_s^i|} \quad (5)$$

If  $\mathcal{C}_r^j > \bar{\mathcal{C}}_s$ , we update all part labels in  $\mathcal{P}_s^j$  to  $\ell_r^j$ . Otherwise, new label categories are assigned to those  $\ell_s^i$  whose  $\mathcal{C}_s^i > \mathcal{C}_r^j$ . For each new label assignment  $\ell_s^i \rightarrow \ell_t$ , we search the corresponding region in  $G_{topK}$  for all Gaussians with  $\ell_r^j$ , and update their labels to  $\ell_t$ .

- **Whole Match:** This is similar to the Partial Match, but the new part labels come from  $\mathcal{L}_r$ . The same comparison process applies.
- **New:** If  $\ell_s^i$  cannot be matched with any label in  $\mathcal{L}_r$ , we assign a new label to this region.

For other cases, we set the labels from  $\ell_s^i$  to the background label (e.g., 0). The *Full Match* is prioritized over other types of matches as it ensures the highest level of consistency.

After label consensus process, we obtain the updated label mapping  $\ell_s \rightarrow \ell_t$  for all labels in  $\mathcal{L}_s$ , and use it to generate the Consistent Input Label Map. We then update the global Label-Class table  $\mathcal{M}_{LC}^G$  based on this mapping and the input  $\mathcal{M}_{LC}^I$ . To enhance consensus accuracy, we also introduce two modules: Input Confidence Update and Part Label Decay.

**Input Confidence Update.** To avoid incorrect updates of partially visible objects, before label consensus, we update the input confidence  $\mathcal{C}_s$  based on object completeness in the current view. Using the input label map and  $\mathcal{C}_s$ , we derive the Input Confidence Map  $\mathcal{I}_s \in \mathbb{R}^{H \times W}$ . During GS Voting, the completeness of each  $\ell_r^j \in \mathcal{L}_t$  is calculated as the ratio of visible Gaussians in the current view to the total number in the map. We then calculate the coverage ratio map  $Cov_r \in \mathbb{R}^{H \times W}$  from the rendered label map. Next, we perform element-wise multiplication between  $Cov_r$  and  $\mathcal{I}_s$  to obtain the Integrated Confidence Map. The updated confidence  $\mathcal{C}_s^i$  for each  $\ell_s^i$  is the average of the corresponding values in the Integrated Confidence Map.

**Part Label Decay.** Several studies [32]–[34] have found that SAM often over-segments objects, which means that a whole object tends to be split into multiple parts. Therefore, we propose a decay mechanism for label confidence. Specifically, whenever a label in a scene is identified as a part label, its confidence is reduced by  $\delta$ . This decay ensures that over-segmented labels will be correctly updated over multiple accurate observations.

#### D. Segmentation Counter Pruning

We observe that in regions with less view constraint, Gaussians can grow excessively large during optimization. This not only prevents the introduction of new Gaussians but also results in inaccurate segmentation. To address this, we propose a Segmentation Counter Pruning method. For fully

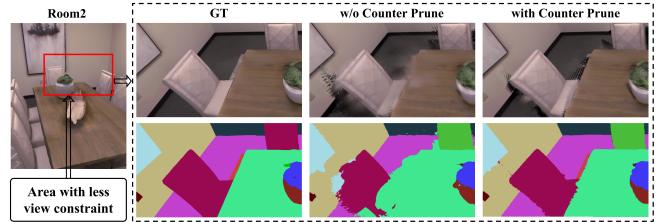


Fig. 3: Effect of Segmentation Counter Pruning. *Left:* We select a region with less view constraint throughout the SLAM process. *Right:* Rendered results from new viewpoints with and without the Segmentation Counter Pruning in this region.

matched labels  $\ell_s^i$  and  $\ell_r^j$  obtained from the label consensus process, we collect all pixels in the symmetric difference between their respective label maps, denoted as  $S$ . Using  $G_{topK}$ , we identify Gaussians in  $S$  with  $\ell_r^j$  as *Counter Gaussians*. If their scale in any direction exceeds a threshold  $\theta$ , they are pruned. This method improves segmentation accuracy and allows for the introduction of new Gaussians that better capture the spatial structure, as shown in Fig.3.

## IV. EXPERIMENTS

Given a real-time RGB-D video stream, our method incrementally fuses 2D semantic information from the 2D foundation models to construct a semantic 3D Gaussian scene representation and tracks itself within the scene. Notably, our method does not require any predefined semantic categories. To comprehensively evaluate performance, we focus on both scene understanding and SLAM performance.

For the scene understanding performance, we consider the following factors:

- **Multi-view 2D Segmentation:** This metric evaluates the consistency and accuracy of 2D segmentation across multiple viewpoints within the scene.
- **3D Object-Level Understanding:** This evaluates the effectiveness of our semantic 3DGS scene representation in achieving 3D object-level scene understanding.

For the SLAM performance, we assess the model’s *camera tracking accuracy* and *reconstruction quality*.

#### A. Experimental Setup

**Datasets:** We evaluate our method on both synthetic and real-world datasets. For scene understanding, we conduct quantitative evaluations on 8 synthetic scenes from Replica [35]. Following other 3DGS-based SLAM methods, we then assess SLAM performance on the Replica [35] and TUM [36] datasets.

**Baselines:** For scene understanding, we first benchmark against existing Semantic SLAM methods on Replica scenes. We also evaluate our model’s capability in open-set scenarios by comparing it with offline methods, including GS-Grouping [28] and Feature 3DGS [26]. For reconstruction quality and camera tracking, we compare our approach against Visual SLAM methods, including SplatAM [10], GS-SLAM [37], LoopSplat [38] and GICP-SLAM [16], as

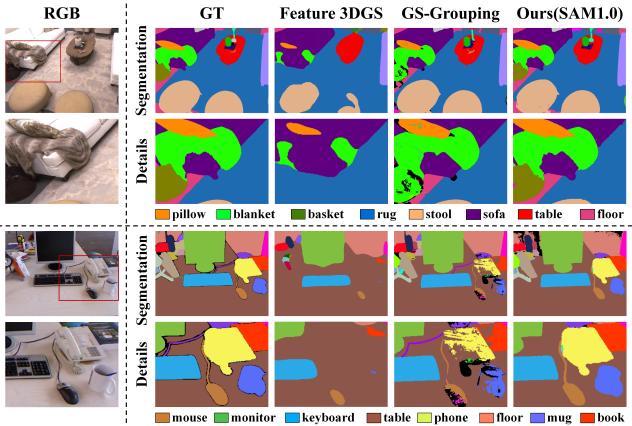


Fig. 4: Qualitative comparison of novel-view **open-set** semantic segmentation. For TUM, novel views refer to viewpoints that are not included in the training data, and the ground truth is obtained from manual annotations.

well as Semantic SLAM methods like NIDS-SLAM [8], DNS-SLAM [9], SNI-SLAM [7], SGS-SLAM [17], NEDS-SLAM [13] and SemGauss-SLAM [12].

**Metrics:** To evaluate scene understanding, we report *mIoU(%)* and assess pixel-wise segmentation accuracy using *Acc(%)*. We also measure the frame rate with *FPS* and consider the model’s complexity by reporting *Learnable Parameters(MB)*. For reconstruction quality, we use standard photometric rendering metrics such as *PSNR*, *SSIM*, and *LPIPS*. Camera tracking is evaluated using *ATE RMSE(cm)*. Notably, for *FPS*, we compare the inference time for both the RGB and segmentation maps, as the baselines employ joint rendering.

**Implementation Details:** Our method is implemented based on GS-ICP SLAM [16]. We also modified the rasterization component in the official 3DGS code [11] to implement GS Voting Splatting. For the 2D Confidence-based Label Consensus, we set the thresholds as  $\tau_1 = 0.85$ ,  $\tau_2 = 0.9$ , and  $\tau_3 = 0.1$ , with the confidence decay parameter  $\delta = 0.06$ . The pruning thresholds  $\theta$  for counter Gaussians are set to 0.10 for Replica and 0.25 for TUM. For the 2D foundation models, we tested RAM++ [18], Yolo-World(v2-I) [20], MobileSAMv2 [15], and SAM1(vit-h) [14], following the default configurations. During experiments, we set the maximum number of labels assignable per test to 2000.

## B. Multi-view 2D Segmentation

We first compare our method to Radiance-Based semantic SLAM approaches on the Replica dataset, following [7]–[9], [13], [17], using semantic priors as input to build the map. As shown in Table I, our model achieves the best performance in semantic segmentation accuracy. We attribute this to our superior semantic scene representation, which integrates explicit semantic information into the 3D Gaussian representation, enabling more precise semantic mapping.

However, these methods are restricted to close-set scene understanding. In contrast, recent offline methods like [26],

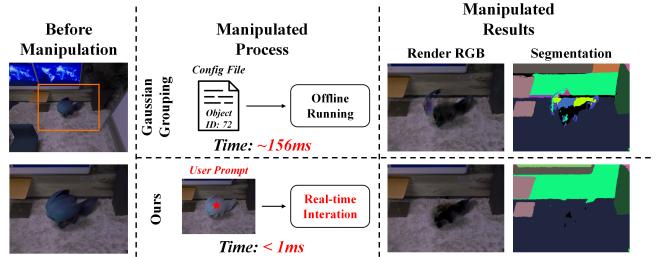


Fig. 5: Scene manipulation process and results. *Left*: We select an object from the scene for removal. *Middle*: Compared to GS-Grouping, our model demonstrates more efficient interactive processing. *Right*: Our method achieves superior manipulation results.

[28] demonstrated strong open-set scene understanding capabilities by using 2D foundation models and incorporating semantic features into 3DGS representation. We evaluate our method’s performance in open-set scenarios by comparing it with these models in zero-shot novel-view semantic segmentation. To ensure fairness, we downsampled keyframe data from our SLAM process by a factor of 10 as input and trained the offline models for 6K iterations.

TABLE I: Quantitative comparison of semantic segmentation accuracy(*mIoU $\uparrow$* ) against Radiance-Based Semantic SLAM methods on the Replica [35] datasets. All models use the ground-truth semantic labels from the replica dataset.

Methods	R0	R1	R2	Of0	Of1	Of2	Of3	Of4
NIDS-SLAM [8]	82.45	84.08	76.99	85.94	-	-	-	-
DNS-SLAM [9]	88.32	84.90	81.20	84.66	-	-	-	-
SNI-SLAM [7]	88.42	87.43	86.16	87.63	78.63	86.49	74.01	80.22
SGS-SLAM [17]	92.95	92.91	92.10	92.90	-	-	-	-
NEDS-SLAM [13]	90.73	91.20	-	90.42	-	-	-	-
Ours(Prior)	<b>93.24</b>	<b>94.11</b>	<b>92.79</b>	<b>93.22</b>	<b>92.16</b>	<b>93.25</b>	<b>93.14</b>	<b>93.77</b>

TABLE II: Comparison of zero-shot novel-view Semantic Segmentation with 3DGS-based **open-set** scene understanding methods. (average performance on Replica [35])

Method	<i>mIoU(%)<math>\uparrow</math></i>	<i>Acc(%)<math>\uparrow</math></i>	<i>Render FPS<math>\uparrow</math></i>	<i>Learnable Parameters(MB)<math>\downarrow</math></i>
Feature 3DGS [26]	48.89	57.51	11.03	894.91
GS-Grouping [28]	59.15	69.94	16.14	717.12
Ours(MobileSAMv2)	57.48	67.14	164.78	314.11
Ours(SAM1.0)	<b>61.91</b>	<b>73.11</b>	<b>165.47</b>	<b>301.71</b>

As shown in Table II, our method (using SAM1.0) significantly outperforms the baselines, with **+13%** and **+3%** *mIoU* improvements over Feature 3DGS and GS-Grouping, respectively. Similar improvements are visible in Fig.4. We attribute this to our efficient 2D Confidence-based Label Consensus, which integrates globally consistent semantic information. Moreover, our method uses **2x fewer** learnable parameters by assigning a 1D *GS Label* to each Gaussian, unlike other methods that use N-channel semantic features. Additionally, our method achieves **10x faster** rendering speeds, owing to the efficiency of our GS Voting Splatting for rapid semantic label map rendering.

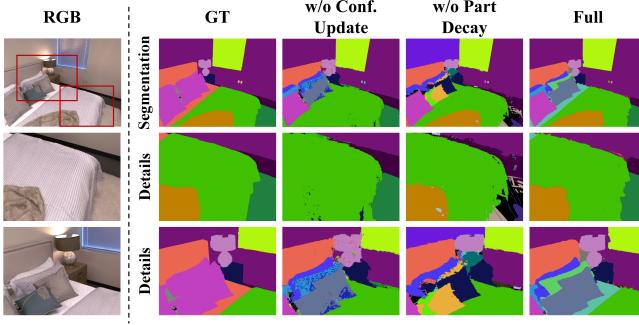


Fig. 6: Visualized ablation results.

### C. 3D Object-Level Scene Understanding

Compared to existing 3DGS-based semantic SLAM methods, our approach not only generates novel-view segmentation results but also supports advanced 3D object-level applications, such as scene manipulation. This includes novel-view synthesis with 3D objects removed, moved, or rotated, as well as interactive scene modifications based on user prompts (e.g., point or bounding box input).

Fig.5 demonstrates that our method achieves more realistic scene manipulation and more efficient user interaction compared to GS-Grouping [28], due to our advanced semantic 3DGS scene representation.

### D. Camera Tracking Accuracy

As illustrated in Table III, our method achieves up to a **51%** relative improvement in tracking accuracy compared to other dense semantic SLAM methods on the Replica dataset. This improvement is attributed to the integration of the G-ICP tracking module derived from GICP-SLAM. Additionally, we also achieve the best localization performance on the TUM dataset, as shown in Table IV. We attribute this improvement to the Segmentation Counter Pruning strategy, which enhances scene representation by refining object contours. As a result, the alignment of Gaussian distributions during the G-ICP process is significantly improved, leading to superior localization accuracy.

### E. Reconstruction Quality

As shown in Table III in the Replica scene, our method achieves the best performance in PSNR and LPIPS compared to other dense semantic SLAM and visual SLAM methods.

TABLE III: Camera Tracking and Reconstruction Results on Replica [35]. (average performance on 8 scenes)

Category	Methods	ATE $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
Visual SLAM	SplaTAM [10]	0.35	33.91	0.969	0.097
	GS-SLAM [37]	0.50	34.27	0.975	0.082
	LoopSplat [38]	0.26	36.63	0.985	0.112
	GICP-SLAM [16]	<b>0.16</b>	<b>38.86</b>	0.976	<b>0.041</b>
Semantic SLAM	SNI-SLAM [7]	0.46	29.43	0.935	0.235
	SGS-SLAM [17]	0.41	34.66	0.973	0.096
	NEDS-SLAM [13]	0.35	34.76	0.962	0.088
	SemGauss-SLAM [12]	0.33	35.03	0.982	0.062
<b>Ours (SAM1.0)</b>		<b>0.16</b>	<b>39.49</b>	0.978	<b>0.034</b>

TABLE IV: Camera Tracking Results on Tum [36]. ATE $\downarrow$

Category	Methods	fr1-desk	fr2-xyz	fr3-office	Avg.
Visual SLAM	SplaTAM [10]	3.35	1.24	5.16	3.25
	GS-SLAM [37]	3.65	-	-	-
	LoopSplat [38]	<b>2.08</b>	1.58	3.22	2.29
	GICP-SLAM [16]	2.41	1.77	2.67	2.28
Semantic SLAM	Ours (SAM1.0)	2.40	<b>1.57</b>	<b>2.48</b>	<b>2.15</b>

### F. Ablation Study.

To better validate our method, we conduct ablation studies on the Replica datasets. We primarily investigate the impacts of our proposed Segmentation Counter Pruning in III-C and the Input Confidence Update, Part Label Decay in III-D. Using our(SAM1.0) as the baseline, we compare segmentation performance after removing each module. The quantitative and qualitative results are presented in Table V and Fig.6, respectively. **1) Part Label Decay:** This module boosted the segmentation results by 5.96 $\uparrow$  mIoU% and 10.2 $\uparrow$  Acc%, as it ensures over-segmented regions are refined after multiple observations, leading to more compact segmentation. **2) Confidence Update:** This module improved results with 3.39 $\uparrow$  mIoU% and 5.4 $\uparrow$  Acc%, as it prevents incorrect updates for partially observed objects during SLAM. **3) Segmentation Counter Pruning:** This module yielded a smaller improvement of 0.26 $\uparrow$  mIoU% and 1.49 $\uparrow$  Acc%, since it primarily refines object boundaries rather than enhancing overall segmentation performance.

TABLE V: Segmentation Performance comparison of various settings on Replica [35].

Settings	mIoU(%) $\uparrow$	Acc(%) $\uparrow$
w/o Segmentation Counter Pruning	62.91	73.02
w/o Input Confidence Update	59.78	69.11
w/o Part Label Decay	57.21	64.31
Ours(SAM1.0)	<b>63.17</b>	<b>74.51</b>

## V. CONCLUSIONS

We propose OpenGS-SLAM, a novel framework that uses 3D Gaussian representation for dense semantic SLAM in open-set scenarios. By integrating semantic labels into the 3D Gaussian model, it enables efficient 3D object-level understanding and dense semantic mapping. We introduce Gaussian Voting Splatting for rapid label map rendering and scene updates. To ensure consistency in multi-view semantic labels from SAM, we propose Confidence-based 2D Label Consensus. Additionally, we implement Segmentation Counter Pruning to refine the scene representation. Extensive experiments show that our approach achieves state-of-the-art performance in tracking, mapping, and scene understanding. However, our method is not applicable to dynamic scenes. In the future, we aim to leverage semantic information to achieve object-level understanding in dynamic environments. We also plan to develop a well-annotated large-scale RGB-D dataset, which includes common and uncommon objects, to further validate the model's performance in open-set environments.

## REFERENCES

- [1] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, “Kimera: an open-source library for real-time metric-semantic localization and mapping,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1689–1696, IEEE, 2020.
- [2] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, “Semanticfusion: Dense 3d semantic mapping with convolutional neural networks,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4628–4635, IEEE, 2017.
- [3] T. Zhang, X. Hu, J. Xiao, and G. Zhang, “A survey of visual navigation: From geometry to embodied ai,” *Engineering Applications of Artificial Intelligence*, vol. 114, p. 105036, 2022.
- [4] Z. Jia, K. Lin, Y. Zhao, *et al.*, “Learning to act with affordance-aware multimodal neural slam,” in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5877–5884, IEEE, 2022.
- [5] Z. Liu, F. Milano, J. Frey, R. Siegwart, H. Blum, and C. Cadena, “Unsupervised continual semantic adaptation through neural rendering,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3031–3040, 2023.
- [6] K. Mazur, E. Sucar, and A. J. Davison, “Feature-realistic neural fusion for real-time, open set scene understanding,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8201–8207, IEEE, 2023.
- [7] S. Zhu, G. Wang, H. Blum, J. Liu, L. Song, M. Pollefeys, and H. Wang, “Sni-slam: Semantic neural implicit slam,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21167–21177, 2024.
- [8] Y. Haghighi, S. Kumar, J. P. Thiran, and L. Van Gool, “Neural implicit dense semantic slam,” *arXiv preprint arXiv:2304.14560*, 2023.
- [9] K. Li, M. Niemeyer, N. Navab, and F. Tombari, “Dns slam: Dense neural semantic-informed slam,” *arXiv preprint arXiv:2312.00204*, 2023.
- [10] N. Keetha, J. Karhade, K. M. Jatavallabhula, G. Yang, S. Scherer, D. Ramanan, and J. Luiten, “Splatam: Splat, track & map 3d gaussians for dense rgb-d slam,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.
- [11] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, “3d gaussian splatting for real-time radiance field rendering.,” *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.
- [12] S. Zhu, R. Qin, G. Wang, J. Liu, and H. Wang, “Semgauss slam: Dense semantic gaussian splatting slam,” *arXiv preprint arXiv:2403.07494*, 2024.
- [13] Y. Ji, Y. Liu, G. Xie, B. Ma, and Z. Xie, “Neds-slam: A novel neural explicit dense semantic slam framework using 3d gaussian splatting,” *arXiv preprint arXiv:2403.11679*, 2024.
- [14] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, “Segment anything,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026, 2023.
- [15] C. Zhang, D. Han, S. Zheng, J. Choi, T.-H. Kim, and C. S. Hong, “Mobilesamv2: Faster segment anything to everything,” *arXiv preprint arXiv:2312.09579*, 2023.
- [16] S. Ha, J. Yeon, and H. Yu, “Rgbd gs-icp slam,” *arXiv preprint arXiv:2403.12550*, 2024.
- [17] M. Li, S. Liu, and H. Zhou, “Sgs-slam: Semantic gaussian splatting for neural dense slam,” 2024.
- [18] X. Huang, Y.-J. Huang, Y. Zhang, W. Tian, R. Feng, Y. Zhang, Y. Xie, Y. Li, and L. Zhang, “Inject semantic concepts into image tagging for open-set recognition,” *arXiv preprint arXiv:2310.15200*, 2023.
- [19] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, *et al.*, “Sam 2: Segment anything in images and videos,” *arXiv preprint arXiv:2408.00714*, 2024.
- [20] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, “Yolo-world: Real-time open-vocabulary object detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16901–16911, 2024.
- [21] S. Liu, Z. Zeng, T. Ren, *et al.*, “Grounding dino: Marrying dino with grounded pre-training for open-set object detection,” *arXiv preprint arXiv:2303.05499*, 2023.
- [22] T. Ren, S. Liu, A. Zeng, *et al.*, “Grounded sam: Assembling open-world models for diverse visual tasks,” *arXiv preprint arXiv:2401.14159*, 2024.
- [23] F. Li, H. Zhang, P. Sun, X. Zou, S. Liu, J. Yang, C. Li, L. Zhang, and J. Gao, “Semantic-sam: Segment and recognize anything at any granularity,” *arXiv preprint arXiv:2307.04767*, 2023.
- [24] M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister, “Langsplat: 3d language gaussian splatting,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20051–20060, 2024.
- [25] J.-C. Shi, M. Wang, H.-B. Duan, and S.-H. Guan, “Language embedded 3d gaussians for open-vocabulary scene understanding,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5333–5343, 2024.
- [26] S. Zhou, H. Chang, S. Jiang, Z. Fan, Z. Zhu, D. Xu, P. Chari, S. You, Z. Wang, and A. Kadambi, “Feature 3dg: Supercharging 3d gaussian splatting to enable distilled feature fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21676–21685, 2024.
- [27] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, “Learning transferable visual models from natural language supervision,” in *International conference on machine learning*, pp. 8748–8763, PMLR, 2021.
- [28] M. Ye, M. Danelian, F. Yu, and L. Ke, “Gaussian grouping: Segment and edit anything in 3d scenes,” in *ECCV*, 2024.
- [29] Y. Wu, J. Meng, H. Li, C. Wu, Y. Shi, X. Cheng, C. Zhao, H. Feng, E. Ding, J. Wang, *et al.*, “Opengaussian: Towards point-level 3d gaussian-based open vocabulary understanding,” *arXiv preprint arXiv:2406.02058*, 2024.
- [30] H. K. Cheng, S. W. Oh, B. Price, A. Schwing, and J.-Y. Lee, “Tracking anything with decoupled video segmentation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1316–1326, 2023.
- [31] Y. Yin, Y. Liu, Y. Xiao, D. Cohen-Or, J. Huang, and B. Chen, “Sai3d: Segment any instance in 3d scenes,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3292–3302, 2024.
- [32] K. J. Oguine, R. D. S. Mukul, N. Drenkow, and M. Unberath, “From generalization to precision: exploring sam for tool segmentation in surgical environments,” in *Medical Imaging 2024: Image Processing*, vol. 12926, pp. 7–12, SPIE, 2024.
- [33] A. Carraro, M. Sozzi, and F. Marinello, “The segment anything model (sam) for accelerating the smart farming revolution,” *Smart agricultural technology*, vol. 6, p. 100367, 2023.
- [34] Y. Liu, S. Halek, L. Li, M. Tomaszewski, S. Wang, R. Baumgartner, J. Yuan, G. Goldmacher, and A. Chen, “Universal 3d ct lesion segmentation using sam with recist annotation,” in *Medical Imaging 2024: Image Processing*, vol. 12926, pp. 23–28, SPIE, 2024.
- [35] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, *et al.*, “The replica dataset: A digital replica of indoor spaces,” *arXiv preprint arXiv:1906.05797*, 2019.
- [36] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of rgb-d slam systems,” in *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 573–580, IEEE, 2012.
- [37] C. Yan, D. Qu, D. Xu, B. Zhao, Z. Wang, D. Wang, and X. Li, “Gs-slam: Dense visual slam with 3d gaussian splatting,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19595–19604, 2024.
- [38] L. Zhu, Y. Li, E. Sandström, S. Huang, K. Schindler, and I. Armeni, “Loopsplat: Loop closure by registering 3d gaussian splats,” 2024.