VLSplat: Vision-Language Guided Object-Centric 3D Gaussian Splatting via Scene Graph

Yuntae Jeon*
Sungkyunkwan University
jyt0131@g.skku.edu

Younho Jeon*
KAIST
117jyh@gmail.com

Sujin Jin
Sungkyunkwan University
kane0113@skku.edu

Sungho Jo†
KAIST
shjo@kaist.ac.kr

Seunghee Park†
Sungkyunkwan University
shparkpc@skku.edu

Abstract

Learning semantic representations for 3D Gaussian Splatting has recently emerged as a promising direction for 3D scene understanding. However, existing 2D-to-3D semantic lifting approaches rely primarily on 2D segmentation supervision and lack object-level priors, causing the learned representation to be driven by local mask evidence rather than holistic object structure. In this paper, we present *VLSplat*, a vision-language guided approach for object-centric semantic lifting in 3D Gaussian Splatting. Our method augments mask-based lifting with object-level semantic priors derived from vision-language models and refines Gaussian representations using an object-group scene graph. The refinement is performed through two language-guided modules: intra-group refinement suppresses semantic noise within object groups, while inter-group acquisition expands object support regions by incorporating structurally relevant candidates from neighboring groups. Extensive experiments on indoor scene datasets demonstrate that *VLSplat* improves the semantic consistency and structural completeness of object-level categories. Our code and models are available at `https://vlsplat.github.io/`.

1 Introduction

Understanding 3D scene semantics from multiple 2D views is a fundamental problem in computer vision, enabling object-level understanding and downstream applications such as robotics [1, 2], AR/VR, scene editing [3, 4], and semantic mesh extraction [5, 6]. Recently, 3D Gaussian Splatting (3DGS) [7] has emerged as a powerful representation for real-time novel-view synthesis, motivating growing interest in learning semantic-aware Gaussian representations for joint rendering and scene understanding. In this setting, the goal is to lift 2D semantic observations, typically segmentation masks, into object-aware 3D Gaussian representations such that 3DGS captures semantic properties of the underlying scene.

Existing semantic 3DGS methods can be broadly divided into language-feature lifting and mask-based grouping. Language embedding approaches learn language latent fields or attach vision-language features to Gaussian primitives for open-vocabulary 3D understanding. Language latent-field methods [8, 9, 10, 11, 12, 13] distill CLIP [14] features into a continuous 3D feature space for natural-language querying, whereas Gaussian-level embedding methods [15, 16] associate such features with individual primitives for more discrete point-level semantics. However, these methods mainly lift or

*Equal contribution.
†Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

associate vision-language semantics with reconstructed Gaussians for retrieval and querying, rather than directly optimizing Gaussian features with vision-language guidance for category-aware object representation learning.

In contrast, mask-based grouping methods use 2D segmentation masks to learn Gaussian groups at instance-level, often producing more explicit object separation than language-field representations [17, 18, 19, 20]. Recently, codebook-based lifting has been introduced for 3DGS semantic lifting, where Unified-Lift [21] associates Gaussian features with a trainable object-level codebook to organize primitives into mask-consistent object regions. However, such methods are induced by local 2D segment masks such as SAM [22], whose view-dependent and multi-level predictions can split the same object into multiple parts or merge semantically different regions into one mask, especially in indoor scenes. As a result, the learned Gaussian groups may appear locally consistent in rendered views, but they are not explicitly grounded in category-level object semantics, limiting complete object-centric representation learning.

In this work, we propose VLSplat, a vision-language guided framework for category-aware object-centric semantic lifting in 3DGS. Our goal is to extend codebook-based Gaussian lifting from category-agnostic instance grouping to category-aware object representation learning under 3DGS optimization. We first construct hybrid 2D supervision by combining text-prompted object masks for major indoor categories with category-agnostic masks for the remaining regions. This provides semantic object cues while retaining dense mask coverage for the scene.

To overcome noise and inconsistency from 2D masks, we reinterpret learned Gaussian-codebook groups as nodes in a scene graph. Each node represents a semantic object candidate or an uncertain fragment, and graph relations provide spatial context for reasoning about whether a group is a meaningful object, noisy component, or candidate region to be reassigned. Based on this graph, VLSplat performs two complementary refinements: intra-group semantic refinement suppresses noisy primitives within object nodes, while inter-group acquisition expands incomplete object support by retrieving structurally plausible candidates from surrounding background regions using vision-language priors.

We conduct extensive experiments on 3D semantic and instance segmentation and point-level semantic segmentation. Following the thing–stuff taxonomy [23], VLSplat targets thing categories, i.e., objects with a well-defined shape such as chairs, desks, and beds. Rather than merely lifting vision-language semantics onto reconstructed Gaussians for retrieval or querying, our method directly optimizes Gaussian features with vision-language guidance through scene-graph structured refinement. Our contributions are summarized as follows:

- We propose *VLSplat*, a vision-language guided semantic lifting framework that extends codebook-based Gaussian grouping from category-agnostic instance clustering to object-centric representation learning.
- We design an object scene graph that interprets codebook groups as nodes enriched with VLM-derived language priors and spatial adjacency, and use it as an object-level prior for Gaussian refinement.
- We introduce language-guided Gaussian gating on synthetic object views, enabling intra-node refinement and inter-node acquisition that improve both the semantic consistency and structural completeness of object groups.

2 Related Work

2D-to-3D Semantic and Instance Lifting. 2D-to-3D lifting transfers 2D masks, segments, or semantic features into 3D fields while enforcing multi-view consistency. Neural-field-based methods lift object selections, panoptic masks, or contrastive segments into volumetric representations [20, 24], while related approaches incorporate open-vocabulary semantics, promptable segmentation, or hierarchical grouping into radiance fields [25, 26, 27]. With the rise of 3DGS, later methods lift 2D mask supervision onto explicit Gaussian primitives for object-level instance segmentation [17, 18, 19, 28, 29, 30]. Unified-Lift [21] further introduces codebook-based Gaussian semantic lifting, where Gaussian features are associated with a trainable object-level codebook to form mask-consistent object regions. Recent works also revisit the lifting solver itself through structured self-training, global label assignment, or modular segmentation on pre-trained 3DGS scenes [31, 32, 33]. However,

existing lifting methods remain strongly tied to 2D mask quality and cross-view consistency, whereas our proposed VLSplat injects object-level language priors and graph-guided reassignment to refine ambiguous or incomplete Gaussian groups.

Vision-Language Gaussian Embedding. Recent works extend 3DGS to language-aware scene understanding by embedding semantic, CLIP [14]-aligned, or foundation-model features into Gaussian representations [9, 11, 12, 13, 34]. These methods distill vision-language features into continuous 3D language fields or Gaussian primitives for open-vocabulary retrieval, natural-language querying, and rendered-view semantic prediction [9, 10, 11, 12, 13, 35, 36]. Recent SAM [22]+CLIP [14] based approaches improve object-level localization by combining object masks with language features: OpenGaussian [16] and LangSplatV2 [15] link SAM-derived instance features with CLIP embeddings for point-level open-vocabulary understanding, and PanoGS [8] uses language-guided graph cuts with SAM-guided affinities for panoptic scene understanding. Beyond per-scene optimization, SceneSplat [37] and Chorus [38] pretrain generalizable 3DGS scene encoders with vision-language or multi-teacher supervision, following self-supervised point representation learning such as Sonata [39]. While effective for queryable semantic fields and transferable scene encoders, these methods mostly treat language features as annotations to be lifted or distilled, whereas VLSplat uses object-level language priors to structurally refine Gaussian object groups.

Structured Scene Graph Reasoning. 3D scene graphs provide structured abstractions for objects, regions, agents, and their relations in 3D environments. Early works construct hierarchical scene graphs for indoor scenes and reconstructed environments, linking object-level semantics with indoor-, or scene-level structure [40, 41, 42]. Later mapping systems extend scene graphs to online spatial perception, dynamic agents, and robot planning through metric-semantic maps and layered graph structures [43, 44, 45]. Recent language-aware scene graphs further support open-vocabulary scene abstraction, open-set relation prediction, language-guided navigation, and context-aware entity grounding [46, 47, 48, 49, 50, 51]. While prior works mostly use scene graphs as output abstractions, structured memories, or downstream reasoning substrates [52, 53, 54, 55], VLSplat uses graph structure as an optimization scaffold for refining object-centric Gaussian groups during semantic lifting.

3 Proposed Approach

VLSplat addresses object-level 3D instance and semantic segmentation in 3D Gaussian Splatting scenes, illustrated in Fig. 1. The method first lifts object-aware 2D semantic masks into 3D with Gaussian-level features and an object-level codebook (Sec. 3.1). It then constructs a 3D scene graph that associates codebook groups with language semantics and spatial context (Sec. 3.2). Finally, VLSplat uses the scene graph as an object-level prior to refine Gaussian features beyond mask-level supervision, producing more object-consistent semantic representations (Sec. 3.3).

3.1 Codebook-based Gaussian Semantic Lifting

Object-level codebook. Object-level 3D segmentation requires grouping dense Gaussian primitives into coherent object regions. To this end, we employ a codebook-based lifting module that learns object-aware Gaussian features from 2D semantic supervision. Given a 3D Gaussian scene $\mathcal{G} = \{g_i\}_{i=1}^{N}$, each Gaussian g_i has a 3D center $\boldsymbol{\mu}_i$, covariance $\boldsymbol{\Sigma}_i$, opacity α_i, color attributes $\mathbf{c}_i$, and a learnable object feature $\mathbf{f}_i \in \mathbb{R}^d$. We jointly learn the Gaussian object features and a compact object-level codebook $\mathbf{F}_{\text{obj}} = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_K]^\top \in \mathbb{R}^{K \times d}$, where K is the maximum number of object-level groups and $\mathbf{p}_k$ denotes the k-th latent group prototype. Each Gaussian is assigned to the prototype with the highest dot-product association score, i.e., $\hat{k}_i = \arg\max_k \text{sim}(\mathbf{f}_i, \mathbf{p}_k)$, where $\text{sim}(\mathbf{f}_i, \mathbf{p}_k) = \mathbf{f}_i^\top \mathbf{p}_k$.

The quality of these initial groups depends heavily on the 2D semantic supervision used for lifting. In indoor scenes, generic mask supervision such as SAM2 [56] often produces view-inconsistent masks or fragmented object regions across views, leading to unstable object grouping in 3D space. To reduce this ambiguity, we construct object-aware semantic maps using SAM3 [57] with category-level text prompts for major indoor objects, and use SAM2 to fill the remaining unlabeled regions, detailed in supplementary. This hybrid supervision combines category-aware masks with dense scene coverage,

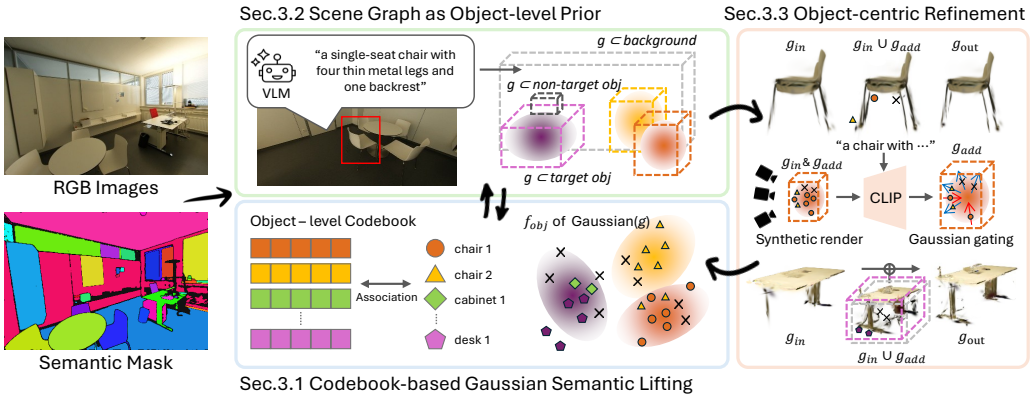


Figure 1: **Overview of VLSplat.** Starting from posed RGB images and object-aware semantic masks, our method learns Gaussian-level semantic features and associates them with an object-level codebook to form object groups. Throughout 3D Gaussian optimization, the semantic representation and scene graph are iteratively updated, allowing object-centric reasoning over Gaussian groups. The scene graph further guides language-based intra-group refinement and inter-group acquisition, improving semantic consistency and structural completeness.

enabling the learned codebook to produce more stable coarse object hypotheses for subsequent scene-graph construction.

3.2 Scene Graph as Object-level Prior

Given the coarse object hypotheses from codebook lifting, we construct an ROI-based scene graph for object-centric refinement. We assign language-aware evidence from training views and augment each node with 3D spatial support. We further use a VLM to convert category-level evidence into object-specific descriptions, providing richer priors for subsequent representation learning.

Spatial object support. Given the codebook assignment $\hat{k}_i$, we collect each codebook group as $G_k = \{i \mid \hat{k}_i = k\}$ and retain its high-confidence support $S_k = \{i \in G_k \mid \langle \bar{\mathbf{f}}_i, \bar{\mathbf{p}}_k \rangle > \tau_{\text{cb}}\}$. Although S_k provides a high-confidence codebook support, feature confidence alone does not ensure spatial coherence. We therefore construct geometry-aware edges over S_k using center distance and anisotropic covariance. For two Gaussians $i, j \in S_k$, an edge is added when their projected supports overlap along the center-to-center direction:

$$(i,j) \in \mathcal{E}_k^{\text{geo}} \quad \text{when} \quad \|\boldsymbol{\mu}_j - \boldsymbol{\mu}_i\|_2 \le \alpha \left(\sqrt{\hat{\mathbf{d}}_{ij}^\top \boldsymbol{\Sigma}_i \hat{\mathbf{d}}_{ij}} + \sqrt{\hat{\mathbf{d}}_{ij}^\top \boldsymbol{\Sigma}_j \hat{\mathbf{d}}_{ij}} \right), \tag{1}$$

where $\hat{\mathbf{d}}_{ij}$ is the normalized direction from $\boldsymbol{\mu}_i$ to $\boldsymbol{\mu}_j$, and α controls the overlap tolerance. The connected components induced by $\mathcal{E}_k^{\text{geo}}$ define spatially coherent candidate supports. After discarding spatially isolated components, we aggregate the remaining components as the spatial support of group k, fit a 3D bounding box to obtain the tight support B_k^{tight} for intra-node refinement, and expand it to derive B_k^{exp} for inter-node refinement.

Language-prior object semantics. After obtaining the spatial support of each codebook group, we assign language-aware semantics and node types using multi-view evidence. Specifically, we project the Gaussian support into the training views and compare the resulting 2D regions with the object-aware semantic maps. We also use a VLM to help interpret the category-level evidence and generate an object-specific description. The resulting description captures visual and structural cues, including shape, parts, material, and contextual appearance, providing object-level language evidence for CLIP [14]-based optimization. We provide an ablation study on the effect of VLM prompts on object representation learning in the supplementary material.

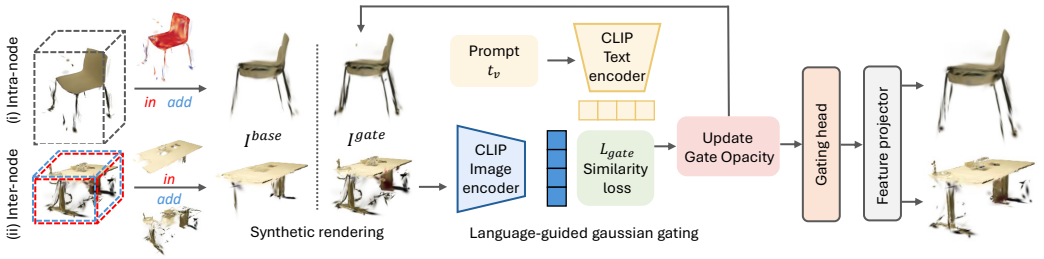


Figure 2: **Vision-language-guided object-centric refinement.** The proposed two-stage refinement consists of **(i) intra-node refinement**, which resolves ambiguous Gaussians within a target node, and **(ii) inter-node acquisition**, which recovers missing object parts from neighboring non-target or background regions.

Scene graph construction. The final scene graph is defined as $\mathcal{H} = (\mathcal{V}, \mathcal{E})$, where each node $v_k \in \mathcal{V}$ represents a semantic object candidate and stores the codebook ID k, semantic category, VLM-generated object description, codebook prototype $\mathbf{p}_k$, tight ROI B_k^{tight}, and expanded ROI B_k^{exp}. Each node also retains SAM3-based target-object evidence from the training views, which indicates whether the ROI consistently corresponds to a meaningful foreground object. The edge set $\mathcal{E}$ encodes neighboring-object relations: two nodes u and v are connected when their expanded ROIs overlap in 3D, i.e., $\text{Vol}(B_u^{\text{exp}} \cap B_v^{\text{exp}}) > 0$. This graph provides two complementary priors for object-centric refinement: **semantic grounding** for interpreting latent codebook groups as object nodes, and **spatial constraints** for restricting ambiguous Gaussian candidates to plausible neighboring object regions.

3.3 Two-stage Object-centric Refinement

Intra-node refinement. The first stage resolves semantic ambiguity within a target node $v \in \mathcal{V}$ of the scene graph $\mathcal{H}$. Although Gaussians assigned to the corresponding codebook are expected to support the same semantic object node, their features may oscillate due to noisy masks, boundary ambiguity, or partial visibility. For the target node v, we collect its assigned Gaussians within the node support as $\mathcal{S}_v = \{i \mid \hat{k}_i = k_v,\ \boldsymbol{\mu}_i \in B_v\}$, and split them into a reliable core and an uncertain candidate set according to feature confidence:

$$\mathcal{G}_{\text{in}}^{\text{intra}} = \{i \in \mathcal{S}_v \mid s_i^{(v)} \geq \tau_f\}, \qquad \mathcal{G}_{\text{add}}^{\text{intra}} = \mathcal{S}_v \setminus \mathcal{G}_{\text{in}}^{\text{intra}}, \tag{2}$$

where $s_i^{(v)} = \langle \bar{\mathbf{f}}_i, \bar{\mathbf{c}}_{k_v} \rangle$ measures the alignment between Gaussian g_i and the prototype of the target node, and τ_f is a confidence threshold. The reliable core provides stable evidence for the target object node, while the uncertain set is further verified before feature reinforcement or suppression.

Inter-node acquisition. The second stage recovers missing parts of the target node $v \in \mathcal{V}$. Some Gaussians that spatially belong to the target object may be background or unlabeled groups because 2D supervision is incomplete or inconsistent across views. For the target node v, we construct an acquisition pool from all Gaussians inside the node support as $\mathcal{T}_v = \{i \mid \boldsymbol{\mu}_i \in B_v\}$. We then define the reliable core and acquisition candidates as

$$\mathcal{G}_{\text{in}}^{\text{inter}} = \{i \in \mathcal{T}_v \mid \hat{k}_i = k_v,\ s_i^{(v)} \geq \tau_f\}, \qquad \mathcal{G}_{\text{add}}^{\text{inter}} = \{i \in \mathcal{T}_v \mid \hat{k}_i \notin \mathcal{X}_v\}, \tag{3}$$

where $\mathcal{X}_v$ denotes the excluded foreground set, including the target node and neighboring foreground nodes that should not be directly absorbed. Unlike intra-node refinement, which verifies uncertain Gaussians already assigned to the target node, inter-node acquisition tests whether non-target or background Gaussians inside the node support provide useful evidence for the target node. This stage improves the semantic completeness of the target node without modifying the underlying geometry.

Synthetic object-view rendering. We evaluate candidate Gaussians $\mathcal{G}_{\text{add}}$ through object-centric rendering. For each target node v, we sample synthetic camera views around its 3D support and render the object from multiple directions. To cover the full object extent, rendering is performed in

an ROI-normalized local coordinate frame. For each sampled view, we render a base image using only the reliable core and a gated image that additionally includes candidate Gaussians:

$$I_v^{\text{base}} = R_v^{\text{norm}}(\mathcal{G}_{\text{in}}), \qquad I_v^{\text{gate}} = R_v^{\text{norm}}(\mathcal{G}_{\text{in}} \cup q \odot \mathcal{G}_{\text{add}}). \tag{4}$$

where $R_{v,m}^{\text{norm}}$ centers and scales the local coordinate frame by the ROI of target node v, allowing each sampled view to capture the full object extent.

Language-guided Gaussian gating. Each candidate Gaussian $g_i \in \mathcal{G}_{\text{add}}$ is assigned a learnable scalar gate $q_i \in [0, 1]$, which controls its opacity contribution during rendering. The gate acts as an evidence estimator: a high value indicates that the candidate supports the target node prompt, while a low value indicates that the candidate is irrelevant or inconsistent. We optimize the gates using CLIP [14] similarity between the rendered object and the node prompt t_v. Instead of maximizing the absolute CLIP score of the gated rendering, we maximize its improvement over the base rendering:

$$\mathcal{L}_{\text{gate}} = -\frac{1}{M} \sum_{m=1}^{M} \left[\text{sim}(I_{v,m}^{\text{gate}}, t_v) - \text{sim}(I_{v,m}^{\text{base}}, t_v) \right], \tag{5}$$

where M is the number of synthetic views. This relative objective is important because the base rendering already contains reliable object evidence; thus, the learned gates focus on candidates that provide additional semantic support beyond the current object core.

Object-level feature update. The optimized gates are used as semantic evidence to refine Gaussian object features without changing the geometry. We update only the feature direction while preserving its original norm. Let $\mathbf{u}_i = \mathbf{f}_i / \|\mathbf{f}_i\|_2$ be the normalized feature direction of Gaussian g_i. For each candidate $g_i \in \mathcal{G}_{\text{add}}$, we define a gate-dependent target direction $\mathbf{t}_i = \bar{\mathbf{c}}_{k_v}$ if $q_i \geq \tau_g$, and $\mathbf{t}_i = -\bar{\mathbf{c}}_{k_v}$ otherwise, where τ_g is the gate threshold. The feature direction is then updated by directional interpolation:

$$\mathbf{u}_i' = \text{norm}\left((1 - \rho_i)\mathbf{u}_i + \rho_i \mathbf{t}_i\right), \qquad \mathbf{f}_i' = \|\mathbf{f}_i\|_2 \mathbf{u}_i', \tag{6}$$

where $\text{norm}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|_2$ and ρ_i is a per-Gaussian interpolation weight that moves the cosine similarity to the target prototype toward a bounded target value, with stronger repulsion for candidates with lower gate values. Thus, high-gate candidates are attracted toward the target-node prototype direction, while low-gate candidates are repelled from it. This updates semantic assignment cues while preserving Gaussian positions and scales. Details on synthetic view sampling, refinement scheduling, and the handling of overlapping ROIs are provided in Sec. C of the supplementary material.

4 Experiments

Experimental setup. We evaluate our method on two complementary protocols summarized in Tab. 1. The first protocol follows the standard 3DGS optimization with pruning and densification. Since the learned representation is explicitly designed for differentiable rendering, evaluating rendered semantic maps allows us to measure the quality of the final 3DGS representation. The second protocol isolates the semantic lifting ability at the 3D point-level. For this evaluation, we freeze the input

Table 1: Statistics of evaluation datasets used in our experiments. Each scene is optimized independently, and Train and Test denote the views of the same scene used for optimization and evaluation.

Protocol	Input Type	Dataset	Domain	Scenes	Average		
					Objects	Train	Test
Render-level	RGB&Sparse points	ScanNet++ [58]	DSLR	10	27	247	17
		Replica [59]	Simulation	8	20	91	9
Point-level	RGB&Dense points	ScanNetV2 [60]	RGB-D	7	7	2965	-
		Replica [59]	Simulation	8	20	400	-

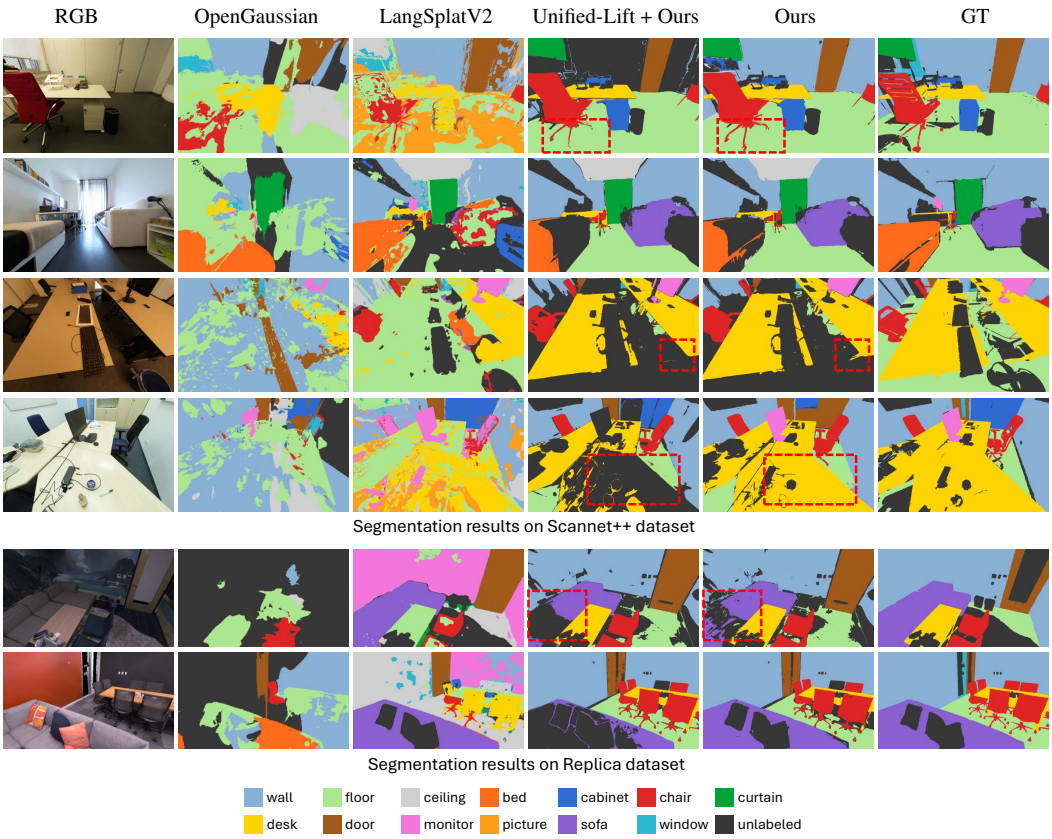


Figure 3: Qualitative render-level 3DGS semantic segmentation results on ScanNet++ and Replica.

3D primitives and disable pruning, densification, and position updates, following the point-level evaluation setting used in previous 3D Gaussian segmentation work [16].

We adopt Unified-Lift [21] as the codebook-based lifting baseline and train it for 15k iterations to obtain coarse object-level Gaussian groups. From 15k to 30k iterations, we update the scene graph every 500 iterations and perform object-level refinement based on the updated graph. At each update, Gaussian gating is optimized for 40 iterations with mini-batches of four rendered views using the CLIP [14] ViT-L model, after which intra-node refinement and inter-node acquisition are applied; full implementation details are provided in Sec. C of the supplementary material. All experiments, including semantic mask preparation, 3DGS training, scene-graph construction, and object-level refinement, are conducted on a single NVIDIA RTX 3090 GPU with 24GB VRAM. Results are reported from a single run per scene.

Thing–stuff taxonomy. Following the thing–stuff taxonomy in panoptic scene understanding [23], we report the metrics separately for our defined thing and stuff splits. Thing categories are bed, cabinet, chair, desk, monitor, and sofa, which serve as target nodes in our scene graph. Stuff categories are wall, floor, ceiling, curtain, door, picture, and window, which are evaluated during rendering but not used as primary targets for object-centric refinement.

4.1 Render-level 3DGS Instance and Semantic Segmentation

Datasets & Evaluation metrics. We evaluate rendered 3D instance and semantic segmentation on ScanNet++ [58] and Replica [59]. We use 10 scenes from ScanNet++ and 8 scenes from Replica. In this protocol, models are trained in the standard 3DGS optimization setting, where densification, pruning, and Gaussian position updates are enabled. We evaluate semantic segmentation quality using mean accuracy (mAcc) and mean intersection-over-union (mIoU).

Table 2: Quantitative 3DGS-based 2D render segmentation results on ScanNet++ and Replica.

Method	ScanNet++				Replica			
	mAcc		mIoU		mAcc		mIoU	
	Thing	Stuff	Thing	Stuff	Thing	Stuff	Thing	Stuff
LegGaussian [11]	27.09	33.73	20.89	13.19	(OOM*)			
OpenGaussian [16]	19.09	37.69	18.33	26.10	12.89	9.32	12.03	7.46
LangSplatV2 [15]	65.69	57.82	61.32	39.94	66.88	56.40	59.64	35.80
Unified-Lift [21] + Ours	64.77	**74.99**	64.42	**68.63**	65.99	67.42	65.93	65.55
Ours	**70.31**	74.09	**69.81**	67.79	**68.69**	**68.11**	**68.59**	**67.75**

Table 3: Quantitative 3D point cloud segmentation results on ScanNetV2 and Replica. All methods are evaluated on the full scene under an identical protocol, except Ours †, which restricts evaluation to scene graph-based ROIs to assess object-level quality.

Method	ScanNetV2				Replica			
	mIoU	mAcc	PRQ(T) ↑	PRQ(S) ↑	mIoU	mAcc	PRQ(T) ↑	PRQ(S) ↑
OpenGaussian [16]	29.73	41.69	-	-	32.56	38.81	-	-
PanoGS [8]	54.50	74.98	35.65	**30.32**	56.91	68.27	35.76	**64.60**
Ours	40.49	59.45	27.27	16.48	64.03	74.34	47.87	37.28
Ours †	**61.89**	**79.12**	**46.18**	27.80	**72.20**	**80.37**	**54.80**	38.21

We compare our method with OpenGaussian [16], LangSplatV2 [15], and Unified-Lift [21]. Since the original Unified-Lift produces unlabeled object groups, "Unified-Lift + Ours" assigns category labels to its groups using our object-aware semantic maps and scene-graph based category association. It shares the same masks and training schedule as VLSplat but omits the iterative scene-graph refinement, allowing a direct comparison of the refinement under identical supervision.

Render-level 3DGS evaluation results. Tab. 2 reports the quantitative results on ScanNet++ and Replica. VLSplat achieves strong performance on both datasets, with particularly consistent gains on Thing-level metrics. This demonstrates the effectiveness of our scene-graph-guided refinement for object-centric regions, where each target node can be grounded by category-aware masks, language descriptions, and local 3D support. For Stuff classes, the gains are smaller and in some cases the performance is slightly lower. This is consistent with the design of VLSplat, which focuses on foreground object nodes rather than large amorphous layout regions. The qualitative results in Fig. 3 further support this trend, showing that VLSplat produces more coherent object regions and better preserves fine-grained object parts. Class-wise metrics are provided in the supplementary material.

4.2 3D Point Cloud Semantic Segmentation

Datasets & Evaluation metrics. Following prior point-level 3D Gaussian segmentation works [16], we evaluate 3D semantic and instance segmentation on ScanNetV2 [60] and Replica [59]. Both datasets provide annotated 3D geometry with point-level semantic and instance labels, allowing predictions to be directly evaluated on fixed 3D support. For semantic segmentation, we report mIoU and mAcc. For instance-aware evaluation, we report panoptic reconstruction quality [61], i.e., PQ [62] with an IoU matching threshold of 0.25, for thing and stuff categories, denoted as PRQ(T) and PRQ(S), respectively.

Point-level 3DGS evaluation results. Tab. 3 reports point-level segmentation results on ScanNetV2 and Replica. Compared with OpenGaussian [16], which assigns features to individual Gaussian primitives, our method benefits from organizing primitives into object-level nodes, leading to stronger semantic assignment on fixed 3D support. On Replica, VLSplat also outperforms PanoGS [8] in both mIoU and mAcc, while PanoGS achieves higher full-scene scores on ScanNetV2. The two methods differ most at the object level (Fig. 4 and Fig. 10 in supp.). PanoGS embeds semantics in a continuous feature field, which tends to merge objects on a desk into the desk and to label the legs of chairs and

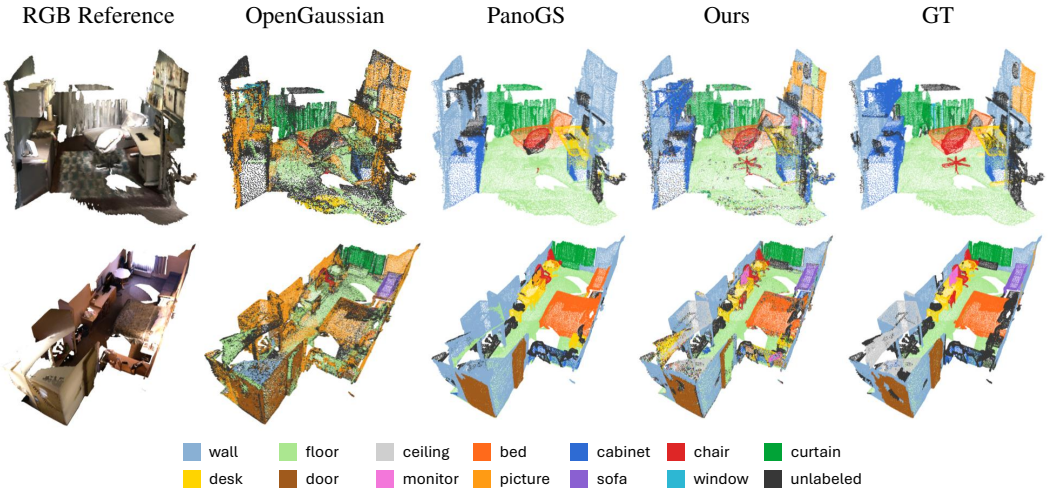


Figure 4: Qualitative 3D point-level segmentation results on ScanNetV2.

Table 4: Scene graph construction ablation on ScanNetV2 for object-level localization accuracy.

Case	Euc.	Aniso.	Feature	mIoU	mAcc
#1				40.49	59.45
#2	✓			42.70	63.64
#3		✓		43.88	64.91
#4			✓	58.51	76.11
#5		✓	✓	**61.89**	**79.12**

Table 5: Intra/inter-refinement ablation on ScanNet++, Thing.

Case	Inter	Intra	mAcc.	mIoU
#1			61.58	61.25
#2	✓		69.85	69.45
#3		✓	69.74	69.31
#4	✓	✓	**70.31**	**69.81**

desks as floor where they meet the ground. VLSplat keeps these as separate, complete segments by grouping Gaussians into discrete object nodes. This object-level quality is reflected in Ours †, which evaluates within scene-graph ROIs and therefore complements, rather than directly compares with, the full-scene results. The relatively lower PRQ(S) suggests that our refinement mainly improves object-centric foreground nodes, while large background or layout regions are less directly refined.

4.3 Ablation Study

Ablation on scene graph. We analyze the effect of scene-graph ROI construction in Tab. 4. All variants are evaluated at the same training iteration, restricting evaluation to points within the resulting ROIs, while #1 evaluates the full scene without ROIs. We compare Euclidean distance, covariance-aware anisotropic connectivity, feature similarity, and their combination for constructing object-level ROIs. Feature similarity provides a large improvement over purely spatial grouping, indicating that semantic consistency is essential for localizing object nodes. Combining feature similarity with anisotropic Gaussian connectivity achieves the best performance, showing that covariance-aware spatial support further improves ROI coherence and object-level localization.

Ablation on object-level refinement. We evaluate the contribution of intra-node refinement and inter-node acquisition in Tab. 5, with the scene graph construction fixed. Each module alone substantially improves the baseline, and combining both achieves the best mAcc and mIoU, although the additional gain over either module alone is modest. While the two modules operate on disjoint candidate sets, their effects overlap in practice: both mainly correct ambiguous Gaussians around object boundaries and thin structures under the same language prior, so once one module resolves the dominant ambiguities, fewer errors remain for the other. The two modules nevertheless address complementary errors, as intra-node refinement removes incorrectly included Gaussians, whereas inter-node acquisition recovers missing object parts such as chair legs.

5 Conclusion

We presented *VLSplat*, a vision-language guided framework for object-centric semantic lifting in 3D Gaussian Splatting. VLSplat addresses the limitations of mask-based lifting by using object-level language priors and scene-graph structure to refine Gaussian features beyond local 2D mask evidence. Through intra-node refinement and inter-node acquisition, our method suppresses semantic noise within object groups and recovers missing object support from neighboring regions. Experiments on render-level and point-level segmentation show that VLSplat improves semantic consistency and structural completeness, particularly for target object categories.

Limitations. Although VLSplat can recover missing object support caused by view-level mask inconsistency, it still requires a reasonable coarse object group to be formed during the initial lifting stage. If an object is rarely observed or rarely detected across the input views, the codebook may fail to construct a reliable object node, and synthetic object-view rendering may provide insufficient visual evidence for CLIP-based gating. Stronger 3D or generative priors, such as SAM3D [5] or video diffusion models, could help complete such weakly observed regions, but they also introduce challenges in multi-view consistency and alignment with the reconstructed 3D coordinate system.

In addition, VLSplat relies on foundation models such as SAM3 [57], a VLM, and CLIP [14], and its sensitivity to these choices has not been systematically studied. Our evaluation also focuses on thing categories in indoor scenes. Finally, the refinement introduces additional offline optimization cost that grows with the number of target nodes.

Broader Impact. VLSplat improves object-centric 3D scene understanding by organizing Gaussian primitives into semantically meaningful object representations. This can benefit downstream applications such as robotics, AR/VR, and semantic mapping. Such representations could support tasks requiring object-level assessment of 3D scenes, such as progress monitoring in built environments [63]. However, inaccurate semantic lifting may lead to incorrect object localization or unreliable scene interpretation, especially in safety-critical applications.

Acknowledgments and Disclosure of Funding

This research was supported by a National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT) [RS-2024-00336270] and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) [RS-2025–02223612].

References

[1] Chenyangguang Zhang, Alexandros Delitzas, Fangjinhua Wang, Ruida Zhang, Xiangyang Ji, Marc Pollefeys, and Francis Engelmann. Open-vocabulary functional 3d scene graphs for real-world indoor spaces. In *CVPR*, 2025.

[2] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, M. Omama, Tao Chen, Shuang Li, Ganesh Iyer, Somya Saryazdi, Nikhil Keetha, Ayush Tewari, Joshua B. Tenenbaum, Celso M. de Melo, K. Madhava Krishna, Liam Paull, Florian Shkurti, and Antonio Torralba. Conceptfusion: Open-set multimodal 3d mapping. In *Robotics: Science and Systems*, 2023.

[3] Jingyu Zhuang, Chen Wang, Liang Lin, Lingjie Liu, and Guanbin Li. Dreameditor: Text-driven 3d scene editing with neural fields. In *SIGGRAPH*, 2023.

[4] Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. Gaussianeditor: Swift and controllable 3d editing with gaussian splatting. In *CVPR*, 2024.

[5] Xingyu Chen, Fu-Jen Chu, Pierre Gleize, Kevin J Liang, Alexander Sax, Hao Tang, Weiyao Wang, Michelle Guo, Thibaut Hardin, Xiang Li, et al. Sam 3d: 3dfy anything in images. In *CVPR*, 2026.

[6] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *CVPR*, 2024.

[7] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 2023.

[8] Hongjia Zhai, Hai Li, Zhenzhe Li, Xiaokun Pan, Yijia He, and Guofeng Zhang. Panogs: Gaussian-based panoptic segmentation for 3d open vocabulary scene understanding. In *CVPR*, 2025.

[9] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *CVPR*, 2024.

[10] Wanhua Li, Renping Zhou, Jiawei Zhou, Yingwei Song, Johannes Herter, Minghan Qin, Gao Huang, and Hanspeter Pfister. 4d langsplat: 4d language gaussian splatting via multimodal large language models. In *CVPR*, 2025.

[11] Jin-Chuan Shi, Miao Wang, Hao-Bin Duan, and Shao-Hua Guan. Language embedded 3d gaussians for open-vocabulary scene understanding. In *CVPR*, 2024.

[12] Xingxing Zuo, Pouya Samangouei, Yunwen Zhou, Yan Di, and Mingyang Li. FMGS: Foundation model embedded 3D gaussian splatting for holistic 3D scene understanding. *International Journal of Computer Vision*, 2025.

[13] 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.

[14] 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.

[15] Wanhua Li, Yujie Zhao, Minghan Qin, Yang Liu, Yuanhao Cai, Chuang Gan, and Hanspeter Pfister. Langsplatv2: High-dimensional 3d language gaussian splatting with 450+ fps. In *NeurIPS*, 2025.

[16] Yanmin Wu, Jiarui Meng, Haijie Li, Chenming Wu, Yahao Shi, Xinhua Cheng, Chen Zhao, Haocheng Feng, Errui Ding, Jingdong Wang, et al. Opengaussian: Towards point-level 3d gaussian-based open vocabulary understanding. In *NeurIPS*, 2024.

[17] Haiyang Ying, Yixuan Yin, Jinzhi Zhang, Fan Wang, Tao Yu, Ruqi Huang, and Lu Fang. Omniseg3d: Omniversal 3d segmentation via hierarchical contrastive learning. In *CVPR*, 2024.

[18] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *ECCV*, 2024.

[19] Weijie Lyu, Xueting Li, Abhijit Kundu, Yi-Hsuan Tsai, and Ming-Hsuan Yang. Gaga: Group any gaussians via 3d-aware memory bank. *arXiv preprint arXiv:2404.07977*, 2024.

[20] Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Buló, Norman Müller, Matthias Nießner, Angela Dai, and Peter Kontschieder. Panoptic lifting for 3d scene understanding with neural fields. In *CVPR*, 2023.

[21] Runsong Zhu, Shi Qiu, Zhengzhe Liu, Ka-Hei Hui, Qianyi Wu, Pheng-Ann Heng, and Chi-Wing Fu. Rethinking end-to-end 2d to 3d scene segmentation in gaussian splatting. In *CVPR*, 2025.

[22] 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.

[23] Gaku Narita, Takashi Seno, Tomoya Ishikawa, and Yohsuke Kaji. Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. In *IROS*, 2019.

[24] Zhongzheng Ren, Aseem Agarwala, Bryan Russell, Alexander G. Schwing, and Oliver Wang. Neural volumetric object selection. In *CVPR*, 2022.

[25] Haoran Chen, Kenneth Blomqvist, Francesco Milano, and Roland Siegwart. Panoptic vision-language feature fields. *IEEE Robotics and Automation Letters*, 2024.

[26] Jiazhong Cen, Zanwei Zhou, Jiemin Fang, Chen Yang, Wei Shen, Lingxi Xie, Dongsheng Jiang, Xiaopeng Zhang, and Qi Tian. Segment anything in 3D with NeRFs. In *NeurIPS*, 2023.

[27] Chung Min Kim, Mingxuan Wu, Justin Kerr, Ken Goldberg, Matthew Tancik, and Angjoo Kanazawa. GARField: Group anything with radiance fields. In *CVPR*, 2024.

[28] Bin Dou, Tianyu Zhang, Yongjia Ma, Zhaohui Wang, and Zejian Yuan. CoSSegGaussians: Compact and swift scene segmenting 3D gaussians with dual feature fusion. *arXiv preprint arXiv:2401.05925*, 2024.

[29] Seokhun Choi, Hyeonseop Song, Jaechul Kim, Taehyeong Kim, and Hoseok Do. Click-Gaussian: Interactive segmentation to any 3D gaussians. In *ECCV*, 2024.

[30] Jens Piekenbrinck, Christian Schmidt, Alexander Hermans, Narunas Vaskevicius, Timm Linder, and Bastian Leibe. Opensplat3d: Open-vocabulary 3d instance segmentation using gaussian splatting. In *CVPRW*, 2025.

[31] Yu Wang, Xiaobao Wei, Ming Lu, and Guoliang Kang. PLGS: Robust panoptic lifting with 3D gaussian splatting. *IEEE Transactions on Image Processing*, 2025.

[32] Qiuhong Shen, Xingyi Yang, and Xinchao Wang. FlashSplat: 2D to 3D gaussian splatting segmentation solved optimally. In *ECCV*, 2024.

[33] Rohan Chacko, Nicolai Häeni, Eldar Khaliullin, Lin Sun, and Douglas Lee. Lifting by gaussians: A simple fast and flexible method for 3D instance segmentation. In *WACV*, 2025.

[34] 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.

[35] Jiaxiong Qiu, Liu Liu, Xinjie Wang, Tianwei Lin, Wei Sui, and Zhizhong Su. GLS: Geometry-aware 3D language gaussian splatting. *arXiv preprint arXiv:2411.18066*, 2024.

[36] Runnan Chen, Xiangyu Sun, Zhaoqing Wang, Youquan Liu, Jiepeng Wang, Lingdong Kong, Jiankang Deng, Mingming Gong, Liang Pan, Wenping Wang, and Tongliang Liu. OVGaussian: Generalizable 3D gaussian segmentation with open vocabularies. *arXiv preprint arXiv:2501.00326*, 2025.

[37] Yue Li, Qi Ma, Runyi Yang, Huapeng Li, Mengjiao Ma, Bin Ren, Nikola Popovic, Nicu Sebe, Ender Konukoglu, Theo Gevers, et al. Scenesplat: Gaussian splatting-based scene understanding with vision-language pretraining. In *ICCV*, 2025.

[38] Yue Li, Qi Ma, Runyi Yang, Mengjiao Ma, Bin Ren, Nikola Popovic, Nicu Sebe, Theo Gevers, Luc Van Gool, Danda Pani Paudel, et al. Chorus: Multi-teacher pretraining for holistic 3d gaussian scene encoding. In *CVPR*, 2026.

[39] Xiaoyang Wu, Daniel DeTone, Duncan Frost, Tianwei Shen, Chris Xie, Nan Yang, Jakob Engel, Richard Newcombe, Hengshuang Zhao, and Julian Straub. Sonata: Self-supervised learning of reliable point representations. In *CVPR*, 2025.

[40] Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R. Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *ICCV*, 2019.

[41] Johanna Wald, Helisa Dhamo, Nassir Navab, and Federico Tombari. Learning 3d semantic scene graphs from 3d indoor reconstructions. In *CVPR*, 2020.

[42] Shun-Cheng Wu, Johanna Wald, Keisuke Tateno, Nassir Navab, and Federico Tombari. Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences. In *CVPR*, 2021.

[43] Antoni Rosinol, Marcus Abate, Yun Chang, and Luca Carlone. 3d dynamic scene graphs: Actionable spatial perception with places, objects, and humans. In *Robotics: Science and Systems*, 2021.

[44] Antoni Rosinol, Arjun Gupta, Marcus Abate, Jingnan Shi, and Luca Carlone. Kimera: From slam to spatial perception with 3d dynamic scene graphs. *The International Journal of Robotics Research*, 2021.

[45] Nathan Hughes, Yun Chang, and Luca Carlone. Hydra: A real-time spatial perception system for 3d scene graph construction and optimization. In *Robotics: Science and Systems*, 2022.

[46] Qiao Gu, Alihusein Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bishal Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, Chuang Gan, Celso M. de Melo, Joshua B. Tenenbaum, Antonio Torralba, Liam Paull, and Florian Shkurti. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *ICRA*, 2024.

[47] Yinan Deng, Jiahui Wang, Jingyu Zhao, Xinyu Tian, Guangyan Chen, Yi Yang, and Yufeng Yue. OpenGraph: Open-vocabulary hierarchical 3D graph representation in large-scale outdoor environments. *IEEE Robotics and Automation Letters*, 2024.

[48] Abdelrhman Werby, Chenguang Huang, Martin Büchner, Abhinav Valada, and Wolfram Burgard. Hierarchical open-vocabulary 3D scene graphs for language-grounded robot navigation. In *ICRA*, 2024.

[49] Sebastian Koch, Narunas Vaskevicius, Mirco Colosi, Pedro Hermosilla, and Timo Ropinski. Open3DSG: Open-vocabulary 3D scene graphs from point clouds with queryable objects and open-set relationships. In *CVPR*, 2024.

[50] Haonan Chang, Kowndinya Boyalakuntla, Shiyang Lu, Siwei Cai, Eric Pu Jing, Shreesh Keskar, Shijie Geng, Adeeb Abbas, Lifeng Zhou, Kostas Bekris, and Abdeslam Boularias. Context-aware entity grounding with open-vocabulary 3D scene graphs. In *CoRL*, 2023.

[51] Sebastian Koch, Pedro Hermosilla, Narunas Vaskevicius, Mirco Colosi, and Timo Ropinski. Lang3DSG: Language-based contrastive pre-training for 3D scene graph prediction. In *3DV*, 2024.

[52] Zhijie Yan, Shufei Li, Zuoxu Wang, Lixiu Wu, Han Wang, Jun Zhu, Lijiang Chen, and Jihong Liu. Dynamic open-vocabulary 3D scene graphs for long-term language-guided mobile manipulation. *IEEE Robotics and Automation Letters*, 2025.

[53] Saumya Saxena, Blake Buchanan, Chris Paxton, Peiqi Liu, Bingqing Chen, Narunas Vaskevicius, Luigi Palmieri, Jonathan Francis, and Oliver Kroemer. GraphEQA: Using 3D semantic scene graphs for real-time embodied question answering. In *CoRL*, 2025.

[54] Sebastian Koch, Johanna Wald, Mirco Colosi, Narunas Vaskevicius, Pedro Hermosilla, Federico Tombari, and Timo Ropinski. Relationfield: Relate anything in radiance fields. In *CVPR*, 2025.

[55] Lianggangxu Chen, Xuejiao Wang, Jiale Lu, Shaohui Lin, Changbo Wang, and Gaoqi He. Clip-driven open-vocabulary 3d scene graph generation via cross-modality contrastive learning. In *CVPR*, 2024.

[56] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[57] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. In *ICLR*, 2026.

[58] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *ICCV*, 2023.

[59] 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.

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

[61] Manuel Dahnert, Ji Hou, Matthias Nießner, and Angela Dai. Panoptic 3d scene reconstruction from a single rgb image. In *NeurIPS*, 2021.

[62] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *CVPR*, 2019.

[63] Yuntae Jeon, Dai Quoc Tran, Khoa Tran Dang Vo, Jaehyun Jeon, Minsoo Park, and Seunghee Park. Neural radiance fields for construction site scene representation and progress evaluation with bim. *Automation in Construction*, 2025.

Supplementary Material

In this supplementary material, we provide the following:

- **SAM2–SAM3 Mask Combination (Sec. A):** Details of the hybrid 2D mask construction strategy, including mask sources, merging rules, post-processing, and qualitative comparisons.
- **Limitations of Prompt-based 2D Mask Generation (Sec. B):** Analysis of view-inconsistent prompt-based masks and their effect on Gaussian semantic lifting.
- **Implementation Details of Scene Graph and Refinement (Sec. C**): ROI expansion, synthetic view sampling, refinement scheduling, handling of overlapping ROIs, hyperparameters, and runtime.
- **Effect of VLM Prompts on Object-level Representation Learning (Sec. D):** Qualitative analysis of object-level normalized rendering and CLIP-based Gaussian gating under different language prompts.
- **Additional Experiment Results (Sec. E):** Additional quantitative and qualitative results, including per-class segmentation metrics, object-level rendering results, and detailed point-level segmentation visualizations.

A SAM2–SAM3 Mask Combination

Mask sources. We use both SAM3 [57] and SAM2 [56] to construct the 2D mask supervision used for Gaussian-level semantic lifting. SAM3 is applied with category-level text prompts and provides semantically labeled masks for major indoor classes. In our implementation, the prompt set includes wall, floor, ceiling, bed, cabinet, chair, curtain, desk, door, monitor, picture, sofa, and window. These masks are useful for maintaining object-level semantic consistency, but they do not cover objects outside the prompt vocabulary and may miss small or weakly visible regions.

SAM2 is used through its automatic mask generator. Unlike SAM3, SAM2 does not require text prompts and usually provides dense instance coverage over the image. However, the resulting masks do not have category labels, and a single semantic object can be split into several small instances. This behavior is problematic for our lifting stage because different parts of the same object may be assigned to different group IDs.

Compared variants. We consider three mask construction variants before applying the merged masks to Gaussian-level lifting. The first variant is **SAM2-only**, where all masks are obtained from the SAM2 automatic mask generator. This gives dense coverage, but often produces fragmented instances around object boundaries, furniture, and small cluttered regions. The second variant is **SAM3-only**, where only text-prompted SAM3 masks are used. This gives more stable semantic regions for prompted categories, but leaves uncovered pixels when objects are small or outside the predefined class set. The third variant is our **SAM3-priority merged mask**, where SAM3 is treated as the main semantic source and SAM2 is used only to fill regions not sufficiently covered by SAM3.

Merging rule. Let $\mathcal{M}_3$ be the set of post-processed SAM3 masks and $\mathcal{M}_2$ be the set of SAM2 masks. We first build a binary occupancy map from SAM3:

$$O_3(x) = \bigvee_{m \in \mathcal{M}_3} m(x), \tag{7}$$

where x denotes a pixel location. For each SAM2 mask $m_i \in \mathcal{M}_2$, we compute the fraction of its area that overlaps with the SAM3 occupancy map:

$$r_i = \frac{|m_i \cap O_3|}{|m_i|}. \tag{8}$$

A SAM2 mask is retained only when it is large enough and is not already explained by SAM3:

$$|m_i| \geq 200 \quad \text{and} \quad r_i \leq 0.3. \tag{9}$$



(a) RGB (b) SAM2-only (C) SAM3-only (d) Merged(Ours)

Figure 5: Qualitative comparison of mask construction variants. SAM2-only produces dense but fragmented instance masks. SAM3-only gives cleaner semantic masks for prompted categories but leaves several small regions uncovered. The merged result keeps SAM3 as the semantic backbone and adds SAM2 masks only in insufficiently covered regions. Cyan boxes show the enlarged desk region used for closer comparison.

Masks smaller than 200 pixels are removed as small fragments. Masks whose overlap ratio with SAM3 is larger than 0.3 are also removed, since they usually correspond to regions already covered by a semantically labeled SAM3 mask. The final mask set is therefore

$$\mathcal{M}_{\text{merge}} = \mathcal{M}_3 \cup \{m_i \in \mathcal{M}_2 \mid |m_i| \geq 200,\ r_i \leq 0.3\}. \tag{10}$$

SAM3 post-processing. Before merging, SAM3 masks are filtered to reduce duplicate or noisy semantic regions. Small SAM3 masks with area below 500 pixels are removed. We also suppress duplicate masks across related classes using mask overlap. This is especially needed for categories such as `desk` and `table`, where the same physical object can be detected by both prompts. After this step, SAM3 masks serve as the semantic backbone for the merged supervision.

Repainting. After filtering, the retained SAM2 masks are combined with the post-processed SAM3 masks and converted into a single 16-bit instance map. When multiple masks overlap, masks are painted in ascending order of confidence score, so that higher-confidence masks overwrite lower-confidence ones. This produces a single valid instance ID for each pixel and avoids ambiguous pixel assignments during lifting.

Qualitative comparison. Fig. 5 shows the difference between the mask variants. SAM2-only covers most visible regions, but the desk area contains many small fragmented instances. SAM3-only produces cleaner category-level regions for large objects such as the wall, curtain, bed, chair, and desk, but several small objects and local regions on the desk remain uncovered. The merged result keeps the main semantic regions from SAM3 while adding SAM2 masks in the regions missed by SAM3. This provides denser supervision without fully replacing the object-level consistency of SAM3.

B Limitations of prompt-based 2D mask generation.

Even with our SAM3 [57]+SAM2 [56] mask generation strategy, prompt-based 2D segmentation can remain unstable across views. This issue is particularly common in indoor scenes, where objects of the same category are spatially adjacent or partially occluded. As shown in Fig. 6, neighboring desks may be merged into a single region in one view while being separated into distinct instances in another. Such view-inconsistent masks provide ambiguous supervision for semantic lifting and can lead to fragmented or incorrectly merged Gaussian groups. These failure cases motivate the need for our ROI-based scene graph, which improves grouping consistency by reasoning over object relationships directly in 3D space.

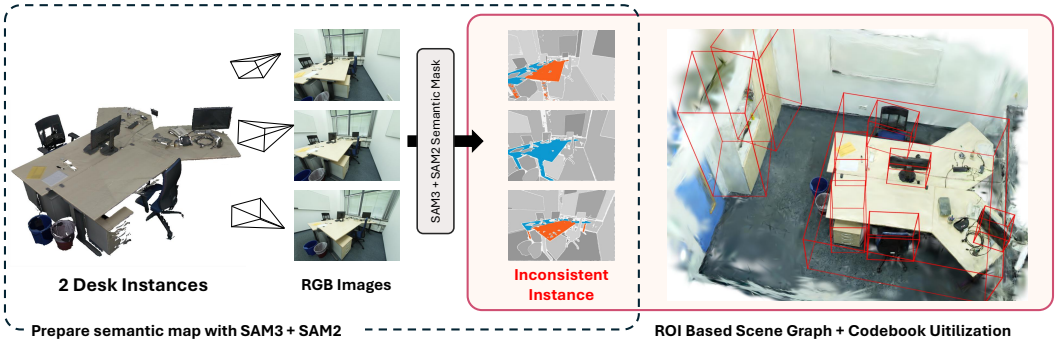


Figure 6: Failure cases of SAM3 under category-level prompting. Despite using the same "desk" prompt across consecutive views, SAM3 produces inconsistent instance-level segmentation results, alternately merging adjacent desks into a single region or separating them into distinct instances.

C Implementation Details of Scene Graph and Refinement

All settings below are shared across datasets without per-scene tuning, and the full configuration is released with the code.

ROI expansion. The expanded ROI B_k^{exp} enlarges the tight ROI B_k^{tight} by 10% of its side length on both faces of every axis with the center fixed. Both ROIs are refit at every graph update.

Synthetic view sampling. Cameras orbit the ROI-normalized object center with azimuth in $[-180°, 180°]$, elevation in $[-30°, 30°]$, and radius in $[0.8, 1.2]$, using a $50°$ field of view and 224×224 resolution. Each gate-optimization iteration uses $M = 4$ views, two sampled views and their azimuth-opposite counterparts, and the CLIP improvement in Eq. (5) is averaged over these views. Each graph update runs 40 iterations, and gates are thresholded at $\tau_g = 0.7$.

Refinement schedule and overlapping ROIs. In each graph update, intra-node refinement is applied to all target nodes, followed by inter-node acquisition. Feature updates are deferred to the end of each update, so all nodes read the same features and the result does not depend on node order. When a Gaussian lies in the ROIs of multiple nodes, their proposed feature directions are averaged into a single update.

Runtime and memory. Tab. 6 reports the cost of each stage on a single NVIDIA RTX 3090. For a scene with about 300 images and 10 target nodes, the full pipeline takes about 146–162 min, roughly 1.5–$1.6\times$ the lifting baseline, with CLIP gating accounting for nearly all of the added cost. Peak memory increases by about 4 GB. At inference, VLSplat adds no graph, CLIP, or VLM computation, so rendering speed matches standard 3DGS.

Table 6: Runtime of each stage.

Stage	Measured rate	Frequency
SAM3 + SAM2 mask preparation	1.8 s / image	Once per view
Codebook lifting	3 min / 1k iterations	3DGS optimization
Node definition	1–2 min / scene	Once
VLM description generation	3–4 s / node	Once
ROI computation	1–2 s / node	Every graph update
Intra-node CLIP gating	4–5 s / node	Every graph update
Inter-node CLIP gating	4–5 s / node	Every graph update

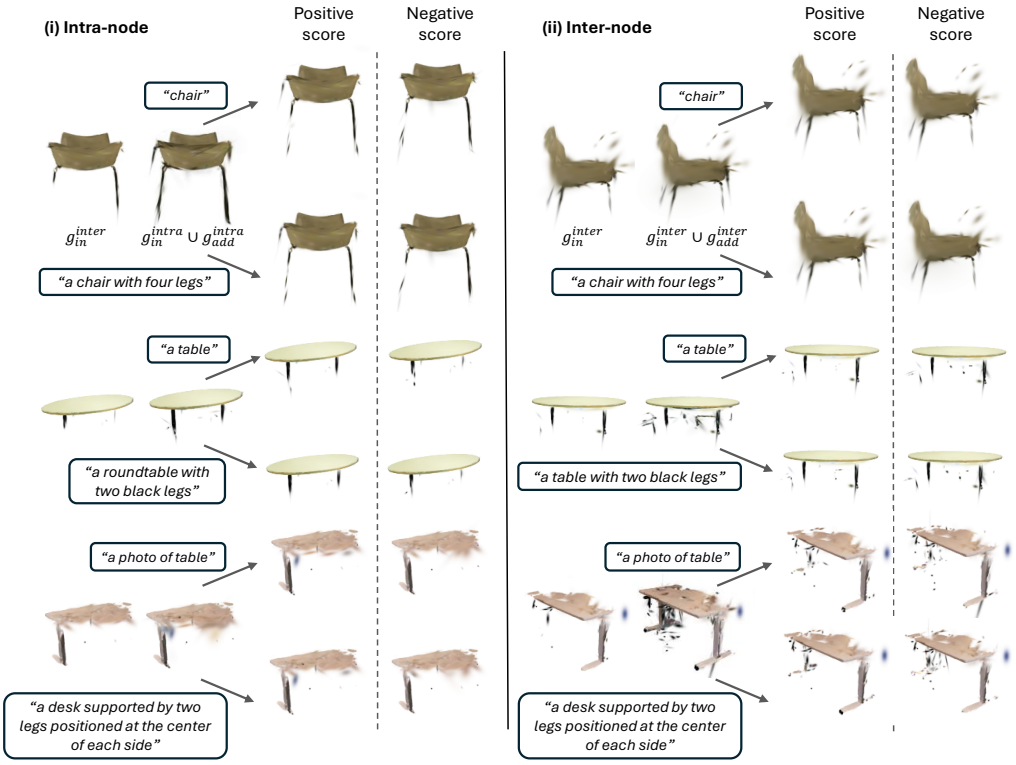


Figure 7: Object-level normalized view rendering and CLIP-based Gaussian gating results on ScanNet++. The first row shows a chair object from scene "8b5caf3398", the second row shows a table object from scene "1ae9e5d2a6", and the last row shows a desk object from scene "67d702f2e8".

D Effect of VLM Prompts on Object-level Representation Learning

Fig. 7 visualizes the proposed intra-node and inter-node refinement process for object-centric Gaussian representation learning. Following Sec. 3.2, we first construct an object scene graph and identify the reliable object Gaussians $\mathcal{G}_{\text{in}}$ and the candidate Gaussians $\mathcal{G}_{\text{add}}$. We then learn the opacity gates of the candidate Gaussians using CLIP [14]-based language guidance. Through this process, intra-node refinement suppresses noisy Gaussians inside the object region while encouraging object-consistent Gaussians to be treated as positive evidence, leading to more discriminative object features. In contrast, inter-node refinement acquires candidate Gaussians from background or unlabeled groups when they are semantically and structurally consistent with the target object.

The results show that the proposed gating mechanism helps recover missing object parts while preventing irrelevant Gaussians from being incorporated. In the first and third rows, inter-node refinement successfully includes thin structural parts such as chair or desk legs, which are often weakly observed in 2D masks or initially assigned to background Gaussian groups. In the second row, although valid table-leg candidates are not sufficiently included in $\mathcal{G}_{\text{add}}$, the learned gate still prevents noisy candidates from being incorrectly merged into the target object. This indicates that the refinement process is useful not only for acquiring missing object parts but also for filtering unreliable candidates.

We further compare different text prompts used for CLIP-based Gaussian gating. The prompts are constructed from object labels obtained from the scene graph and SAM3 masks, and are compared with more general prompt templates such as "a photo of a {class}". Overall, more descriptive prompts that include color or structural attributes provide stronger language guidance for selecting appropriate Gaussians. For example, in the first row, the prompt containing the phrase "four legs" assigns higher positive scores to Gaussians corresponding to the chair legs. Similarly, in the table and desk examples, structure-aware prompts encourage the gating module to assign higher scores to Gaussians that

Table 7: Per-class quantitative segmentation results on ScanNet++ focus-object classes.

Method	ScanNet++											
	Acc						IoU					
	Bed	Cabinet	Chair	Desk	Monitor	Sofa	Bed	Cabinet	Chair	Desk	Monitor	Sofa
LegGaussian	43.61	0.12	5.96	71.09	0.00	41.78	37.62	0.12	5.96	40.16	0.00	41.51
OpenGaussian	9.69	18.91	29.50	23.57	17.74	15.12	8.99	18.65	26.77	23.04	17.46	15.05
LangSplatV2	54.22	51.10	70.11	62.90	**78.79**	77.02	48.63	50.73	65.32	58.61	**71.90**	72.73
Unified-Lift	59.08	55.06	**77.10**	80.77	26.29	90.32	59.06	54.99	**76.55**	**79.54**	26.21	90.21
Ours	**62.48**	**59.78**	77.47	**81.26**	50.00	**90.88**	**62.45**	**59.62**	76.88	79.59	49.58	**90.73**

Table 8: Per-class quantitative segmentation results on Replica focus-object classes.

Method	Replica											
	Acc						IoU					
	Bed	Cabinet	Chair	Desk	Monitor	Sofa	Bed	Cabinet	Chair	Desk	Monitor	Sofa
OpenGaussian	6.45	41.07	19.75	4.99	0.00	5.07	4.25	38.48	19.40	4.99	0.00	5.07
LangSplatV2	5.03	**94.04**	48.91	62.77	**98.06**	**92.46**	3.29	**71.27**	48.31	60.50	**84.19**	**90.26**
Unified-Lift	**56.91**	32.99	**89.81**	**89.96**	73.94	52.35	**56.91**	32.96	**89.63**	**89.82**	73.94	52.34
Ours	**56.50**	33.77	**89.88**	**89.59**	70.16	72.24	**56.50**	33.73	**89.62**	**89.37**	70.16	72.16

better match the target object structure. These qualitative results suggest that detailed VLM prompts improve CLIP-based Gaussian gating and contribute to more complete object-centric representation learning.

E Additional Experiment Results

Quantitative results on class-wise render-level 3DGS semantic segmentation Tab. 7 reports per-class IoU and accuracy results for Thing categories, including bed, cabinet, chair, desk, monitor, and sofa. On ScanNet++, our method achieves the best performance for most categories. Results on Replica are additionally reported in Tab. 8. Although previous approaches also perform competitively on Replica due to its clean indoor scenes, our method performs on par with Unified-Lift on chair and desk, which are among the most dominant objects in indoor environments, and substantially improves sofa. Consequently, our method achieves the best overall Thing-category performance.

Qualitative results on render-level 3DGS segmentation Fig. 8 and Fig. 9 show object-level segmentation visualizations obtained by activating only the Gaussian groups associated with target categories. From top to bottom, we visualize desk, bed, sofa, chair, desk+cabinet, and desk+chair. Compared with the baseline, our method produces cleaner object-level renderings with fewer noisy or unrelated Gaussian regions.

These results indicate that our method organizes 3DGS features into more consistent object-centric groups and improves the grouping of Gaussians themselves, rather than only producing cleaner 2D

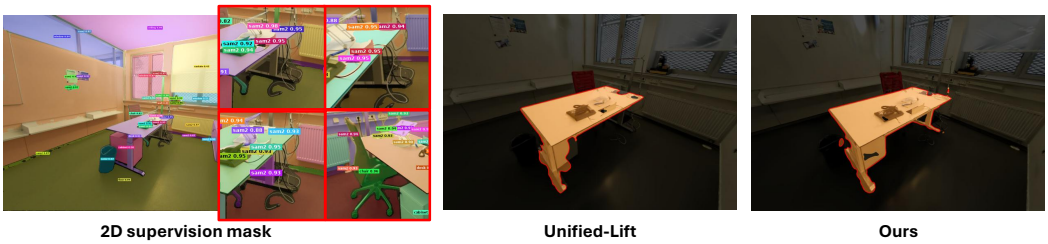


Figure 8: Qualitative results on language-guided object-centric representation learning from inconsistent 2D supervision on Scannet++ scene.

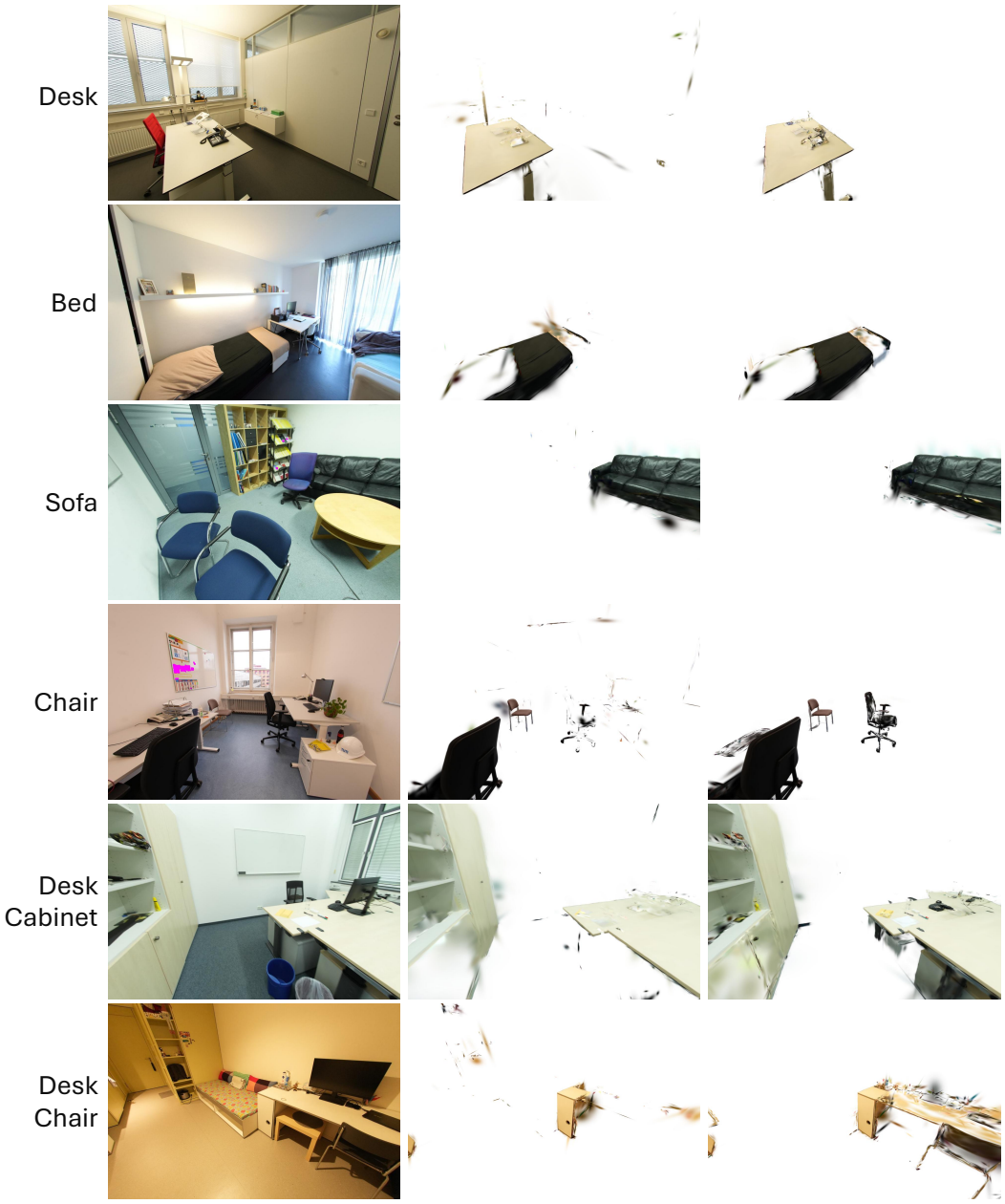


Figure 9: Example of object-level segmentation visualization results. Our method produces cleaner object-centric renderings with fewer fragmented or noisy regions across diverse object categories and multi-object queries.

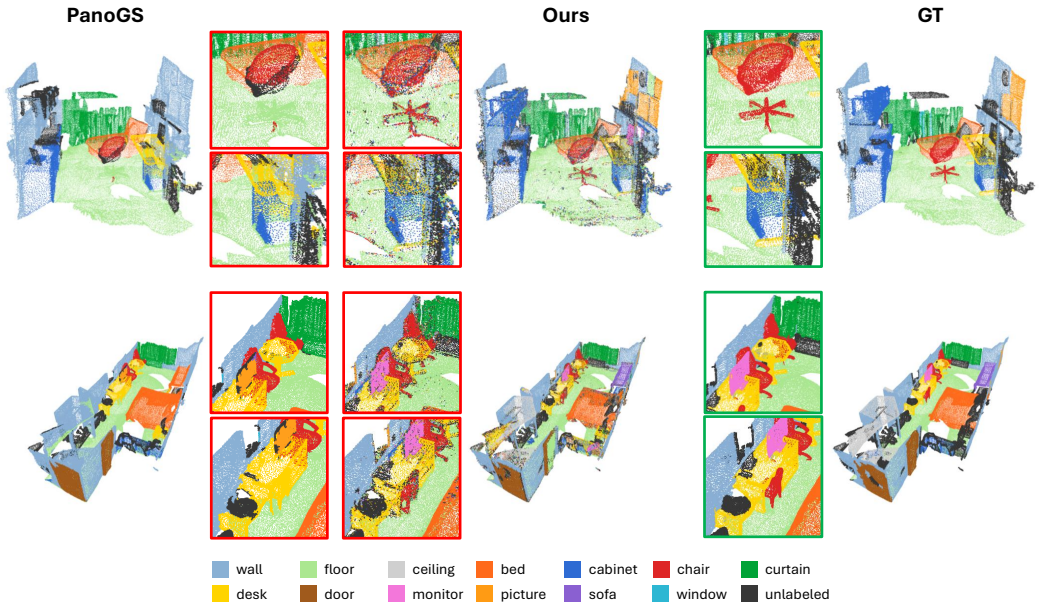


Figure 10: Detailed views of 3D point-level segmentation results on ScannetV2 from 'scene0070' and 'scene0590'. Object regions are presented to provide a clearer comparison of object-level segmentation quality.

semantic renderings. This property can be beneficial for downstream scene understanding tasks, such as Gaussian extraction and object-level editing.

Qualitative results on point-level 3DGS semantic segmentation Fig. 10 provides close-up views of selected object regions in 3D scenes. These visualizations demonstrate the effectiveness of our refinement strategy for 3D point cloud segmentation. In the upper scene, our method produces a cleaner segmentation of the red chair. By performing intra-node refinement, it filters noisy points and incorrectly assigned regions within the object group, reducing mixed-class predictions inside the object and improving object-level separation. Furthermore, inter-node refinement helps recover object parts that are weakly observed by 2D masks. For example, the chair legs are partially assigned to background Gaussian groups in the initial representation, but our method successfully incorporates them by leveraging language-prior information during Gaussian semantic representation learning. As a result, the final segmentation better preserves the structural completeness of the object.

A similar trend is observed in the lower scene. Our method more clearly separates neighboring objects within the same space while preserving fine-grained structures such as thin chair legs. These results indicate that the proposed refinement process improves both local semantic consistency and object-level structural completeness.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer:[Yes]

   Justification: The abstract and introduction state the main contribution of our method, including scene graph-based object-level priors and two-stage object-centric refinement. The claims are supported by quantitative and qualitative results in the experiments.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The Conclusion includes a Limitations paragraph discussing sensitivity to text-prompted semantic mask quality and prompt choices, as well as scalability concerns for more complex scene.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present theoretical results, theorems, or formal proofs. The mathematical formulations are used to describe the proposed method and optimization objectives.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper utilized open-datasets and benchmark evaluation protocols. We also included experimental setup, and training parameter needed to reproduce the main experimental results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The datasets used in this paper (ScanNet++, ScanNetV2, and Replica) cannot be redistributed due to their license terms, but they are publicly available from their original sources after agreeing to the respective terms of use. Instead, we release our implementation code on github.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper provides the experimental protocols, training parameters, evaluation metrics, and compute setup needed to understand the reported results. Additional implementation details are provided in the supplementary material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The paper does not report error bars or statistical significance tests due to the computational cost of repeated scene-level 3DGS optimization. We instead report results across multiple scenes and datasets and provide ablation studies for the main components.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All experiments are conducted on a single GPU, as described in Sec. 4, and the runtime and memory cost of each stage are reported in Sec. C of the supplementary material.

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research uses public datasets and does not involve human-subject experiments, private data, or sensitive personal information. The submission is anonymized and follows the NeurIPS Code of Ethics.

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses potential positive impacts of improved 3D scene understanding for robotics, AR/VR, and semantic mapping. It also notes possible negative impacts, as inaccurate semantic lifting may lead to incorrect object localization or unreliable scene interpretation in safety-critical applications.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release high-risk models or scraped datasets. It uses publicly available indoor scene datasets and does not require additional safeguards for asset release.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper credits the original creators of the datasets, models, and baselines used in our experiments, including ScanNetV2, ScanNet++, Replica, CLIP, SAM3, OpenGaussian, PanoGS, and Unified-Lift. We follow the corresponding licenses and terms of use for research-only evaluation.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper introduces implementation code for the proposed method, and documentation for environment setup, data preparation, training, and evaluation is provided with the released assets.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments, human-subject studies, or participant data collection.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing, human-subject studies, or participant data collection, so IRB approval or equivalent review is not applicable.