NestSplat: Hierarchical Relational Reasoning Segmentation in 3D Gaussian Splatting

Anonymous authors
Paper under double-blind review

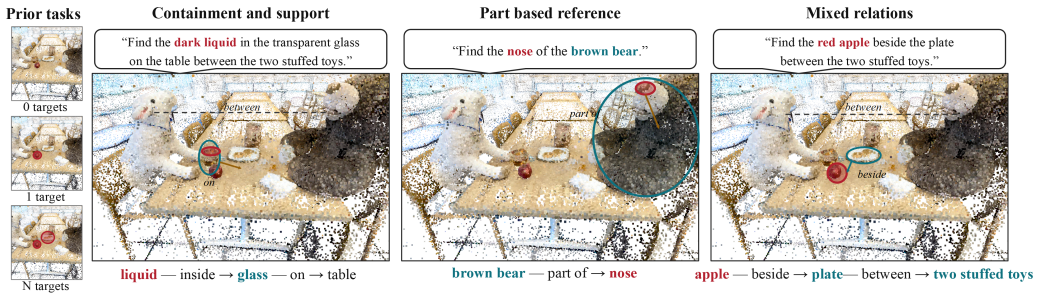


Figure 1: Scope of hierarchical referring in NestSplat. Columns show target cardinality (zero, single, and grouped multiple targets), containment and support, part-based referring, and mixed-relation queries; colors and relation chains denote targets, landmarks, and constraints.

Abstract

Language-guided segmentation in 3D Gaussian Splatting (3DGS) must resolve spatial and relational constraints, yet existing benchmarks emphasize single targets and shallow relations. We introduce Nest-LERF and Nest-ScanNet under zero-target, single-target, and grouped-multiple-target settings. Nest-LERF covers nesting, stacking, part–whole, and mixed relations, while room-scale Nest-ScanNet covers nesting, part–whole, spatial-directional, and mixed relations. We also propose NestSplat, a training-free framework that derives a scene vocabulary from training views, lifts 2D masks into language-anchored 3D supports, and selects query targets through bounded vision–language decisions with geometric context. Without per-scene training, NestSplat achieves 34.16% and 27.42% mIoU on Nest-LERF and Nest-ScanNet, respectively, and transfers strongly to Ref-LERF and Causal-LERF.

1 Introduction

Three-dimensional Gaussian Splatting (3DGS) (Kerbl et al., 2023) has become a prominent representation for 3D reconstruction and embodied perception because it combines real-time rendering with explicit, editable scene primitives. Compared with implicit radiance fields such as NeRF (Mildenhall et al., 2020), 3DGS offers a practical substrate for interactive 3D environments. Lifting semantic priors from 2D vision–language models into Gaussians has further enabled open-vocabulary scene understanding and language-guided object segmentation. Natural instructions, however, are rarely simple noun lookups: they often encode spatial constraints, commonsense, and functional intent. Reasoning-oriented systems such as CausalSplat (Ding et al., 2026) therefore extend language-guided segmentation from direct reference toward relational and functional reasoning.

Current benchmarks still underrepresent fine-grained object-level hierarchy. For tabletop scenes, we organize these constraints into four types: **(1) nesting (containment)**, where a small object lies inside a larger container, e.g., "find the objects stacked in rows inside the transparent box behind the white table"; **(2) stacking (support)**, where vertical support order identifies the target, e.g., "find the silver object beneath the plush rabbit's ears on the white table"; **(3) part–whole relations**, where a whole

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

is first located and a component is then identified, e.g., "find the nose on the toy bear's face"; and **(4) mixed multi-object relations**, where several constraints and distractors occur together, e.g., "on the white table, find the green object between the black camera and the red chair, rather than the green apple." These queries require relational disambiguation rather than category matching alone.

Ref-LERF (He et al., 2025) mainly evaluates noun-level single-target reference, while Causal-LERF emphasizes functional and causal descriptions of individual objects with short relation chains. Multi3DRefer (Zhang et al., 2023) addresses zero, single, and multiple target grounding with 3D boxes; our benchmarks evaluate hierarchical constraints with 2D and 3D masks. For example, if an instruction asks for "the soap dispenser to the left of the sink and the brush above it" but no such configuration exists, the correct output is an empty mask rather than the closest object. We therefore introduce **Nest-LERF**, with 247 queries over four real LERF scenes evaluated by test-view pixel IoU, and **Nest-ScanNet**, with 109 instructions over ten ScanNet scenes evaluated by point-level instance IoU. Nest-LERF uses the four tabletop categories above; Nest-ScanNet retains nesting, part-based, and mixed queries but replaces contact-based stacking with spatial-directional relations suited to room-scale scenes. Queries are annotated as zero-target, single-target, or grouped-N; grouped-N merges same-description instances represented by one noun-anchored support, while heterogeneous multi-node composition remains outside the present scope.

Results on these benchmarks show that existing methods degrade under hierarchical constraints. Open-vocabulary and referring approaches largely reduce an instruction to category matching, whereas current reasoning systems retain a predominantly single-target output space. A natural alternative is to construct a 3D scene graph and delegate unrestricted inference to a language model, but two difficulties arise in 3DGS. First, existing pipelines either optimize semantic features per scene or allow evaluation text to influence lifting, obscuring the contribution of later reasoning stages. Second, unbounded graph inference grows with scene size, while spurious multi-view adjacency can accumulate along a reasoning path. We address both issues with **NestSplat**: a training-free framework that builds language-anchored 3D supports under an object-noun protocol independent of evaluation queries, then performs bounded vision–language selection with component-aware geometric context. The same representation can therefore be applied to a new scene without per-scene semantic training.

Our contributions are fourfold: (1) we formulate hierarchical relational segmentation for 3DGS across dataset-specific containment, support or directional spatial relations, part–whole structure, and mixed constraints under zero-target, single-target, and grouped-N settings; (2) we introduce Nest-LERF and Nest-ScanNet for pixel-level and point-level evaluation; (3) we propose NestSplat, which combines query-decoupled support lifting, bounded vision–language decisions, and component-aware geometry without per-scene semantic training; and (4) NestSplat reaches 34.16% scene-macro mIoU on Nest-LERF, 8.13 points above CausalSplat, and 27.42% on Nest-ScanNet, with additional transfer to Ref-LERF and Causal-LERF.

2 Related Work

3D Gaussian Splatting. 3DGS (Kerbl et al., 2023) represents scenes with explicit Gaussian primitives and differentiable rasterization for high-quality real-time rendering. Mip-Splatting (Yu et al., 2024), 2DGS (Huang et al., 2024a), and Scaffold-GS (Lu et al., 2024) further improve anti-aliasing, geometry, and representation efficiency. Because individual primitives remain explicit and editable, 3DGS provides a structured substrate for semantic annotation and language-guided segmentation.

Language-guided 3D scene understanding. CLIP and 2D masks support open-vocabulary 3D methods (Radford et al., 2021; Kirillov et al., 2023; Ren et al., 2024). Point and radiance-field approaches include OpenScene, SPIn-NeRF, SA3D, and OpenNeRF (Peng et al., 2023; Mirzaei et al., 2023; Cen et al., 2023; Engelmann et al., 2024). Gaussian methods distill features or associate masks with primitives (Qin et al., 2024; Zhou et al., 2024; Wu et al., 2024; Li et al., 2025; Kim et al., 2025; Marrie et al., 2025; Guo et al., 2026; Ye et al., 2024; Choi et al., 2024; Qu et al., 2024); SAGA and GaussianCut add promptable segmentation (Cen et al., 2025; Jain et al., 2024). FlashSplat (Shen et al., 2024) supplies optimal contribution-based mask assignment without per-scene semantic training. ReferSplat (He et al., 2025) targets one support; GRES (Liu et al., 2023) and Multi3DRefer (Zhang et al., 2023) address variable cardinality in 2D masks and 3D boxes, respectively. Scene graphs and reasoning methods model relations or affordances (Gu et al., 2024; Koch et al., 2024;

Xie et al., 2026; Ding et al., 2026; Shi et al., 2026; Wei et al., 2025). NestSplat instead evaluates mask-level hierarchical constraints with zero-, single-, and grouped-target output.

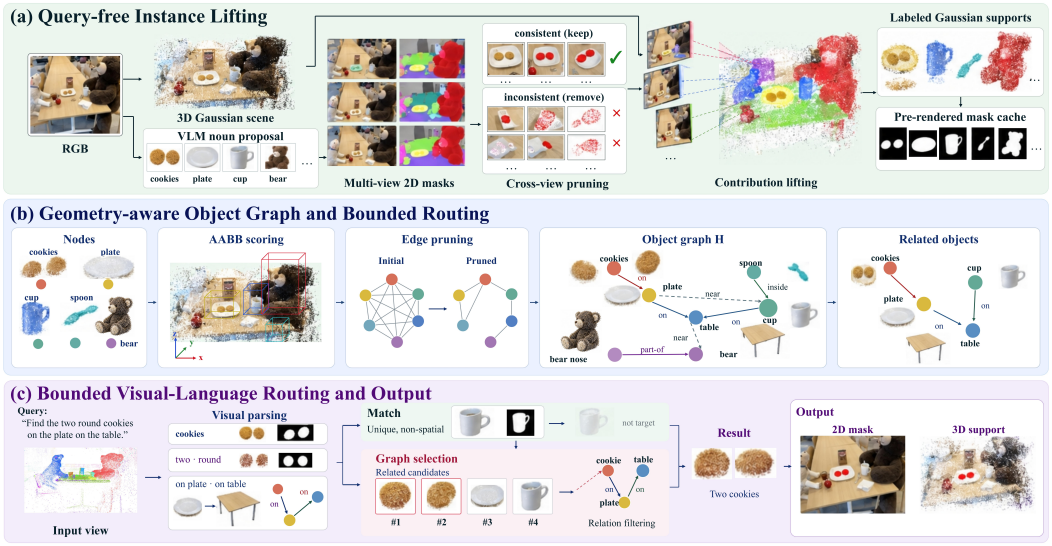


Figure 2: Overview of the three-stage NestSplat framework: scene-level instance lifting, a geometry-aware object graph, and bounded vision-language routing.

3 Method

3.1 Problem Formulation and Overview

Formally, given a 3D Gaussian field $\mathcal{G}$, multi-view images $\mathcal{I} = \{I_i\}_{i=1}^{F}$, and a free-form query s, the task assigns a binary label to every Gaussian and outputs a mask $\hat{\mathbf{M}}_s$. Unlike standard referring segmentation, the task allows no target, one physical target, or a grouped set of same-description physical targets, whose masks are merged for evaluation. Our router returns either no node or one inventory support; a grouped-N output arises when that support covers multiple disconnected same-description instances. Heterogeneous multi-node composition is outside the task scope. The taxonomy is detailed in Section 4.1 and Appendix A.1.

Unlike methods that optimize semantic features for every scene, we attach no learnable features to $\mathcal{G}$ and perform no scene-level training. Language grounding is obtained entirely by training-free lifting of 2D open-vocabulary masks. The lifting vocabulary consists of scene-level object nouns inferred from training observations and is independent of the evaluation-query set $\mathcal{S}$, which we denote by $\mathcal{O} \perp \mathcal{S}$. During scene preparation, this protocol uses only scene observations to generate object nouns, allowing query routing to be evaluated over a fixed lifted representation.

We introduce a three-stage framework that connects unstructured text to language-anchored 3D supports (Figure 2). First, we propose object nouns from training views, generate 2D masks, lift their contributions to per-Gaussian labels, and pre-render a support mask for each test viewpoint (Section 3.2). Second, we export queryable inventory nodes, establish inter-node relations from component-aware geometric proxies, and use a bounded vision–language decision to select the target node v^* at query time (Section 3.3). Finally, 2D evaluation directly retrieves the pre-rendered mask of v^*, whereas 3D evaluation compares its support with point-level ground truth (Section 3.4). Query inference uses at most one image-conditioned call and one inventory-selection call; graph construction invokes no language model.

3.2 Scene-Level Instance Lifting

This module instantiates visible objects from training observations as language-anchored 3D supports without reading any evaluation query. Here, *instance lifting* names the stage; its output is a noun-

conditioned support and is not assumed to correspond one-to-one with a physical instance. Driving segmentation directly with full instructions introduces noise from referring expressions; we therefore extract concise object nouns before lifting them into the Gaussian field.

Object-noun proposal. A vision-language model identifies visible objects in the training observations $\mathcal{I}_{\text{train}}$, yielding a noun set $\mathcal{O} = \{o_i\}_{i=1}^{M}$. Near-synonymous names are merged, while candidate nouns with distinct appearances or categories are retained. Evaluation queries are never read: LERF excludes test views, whereas ScanNet uses all RGB frames because it has no separate test split; in both cases, $\mathcal{O} \perp \mathcal{S}$ is maintained.

2D masks and contribution-based lifting. For each o_i, an open-vocabulary segmentation model generates masks $M_{i,f}$ across multiple views. Nouns without stable 2D evidence are excluded from the inventory. To assign 2D masks to Gaussians, we follow FlashSplat (Shen et al., 2024) and accumulate volumetric-rendering contribution weights. Along ray r, the contribution of Gaussian j is

$$w(r, g_j) = T(r, g_j)\,\alpha(r, g_j), \qquad T(r, g_j) = \prod_{k<j}(1 - \alpha(r, g_k)). \tag{1}$$

Projecting g_j onto the pixel set $\mathcal{R}_f$ in each training view, we accumulate its contributions where the mask is foreground ($k = 1$) or background ($k = 0$):

$$W_k(g_j; o_i) = \sum_{I_f \in \mathcal{I}_{\text{train}}} \sum_{r \in \mathcal{R}_f} \mathbb{I}\big(M_{i,f}(r) = k\big)\, w_f(r, g_j). \tag{2}$$

We label g_j as foreground for o_i if $W_1(g_j; o_i) > W_0(g_j; o_i)$, and as background otherwise.

Cross-view background pruning. Mask noise can incorrectly label background Gaussians as foreground. For a currently foreground Gaussian g, let $N_{\text{val}}(g)$ be the number of views in which it projects into the image and $N_{\text{conf}}(g)$ the number of projections falling on mask background. We define the conflict ratio as

$$\rho_g = \frac{N_{\text{conf}}(g)}{N_{\text{val}}(g) + \varepsilon}. \tag{3}$$

If ρ_g exceeds τ_{bg}, we relabel the Gaussian as background. The refined support is denoted by $\mathcal{P}_i$.

Lifting outputs. The lifting stage computes no node-level geometric parameters. It outputs an object-noun inventory, multi-view 2D masks for each noun, discrete foreground/background memberships over Gaussians, and pre-rendered masks for each retained noun at test viewpoints (Section 3.4). Gaussian primitives store only discrete memberships rather than continuous semantic features; centroids and bounding boxes are parameterized later during node export.

3.3 Geometric Relations and Bounded Routing

The lifted supports are then exported as queryable inventory nodes, with each retained noun inducing one node:

$$v_i = \Big(\mathcal{P}_i,\ \{(\bar{\mu}_{ik}, B_{ik}, w_{ik})\}_{k=1}^{K_i},\ o_i\Big), \tag{4}$$

where o_i is the language anchor. We partition the Gaussian means in $\mathcal{P}_i$ with a local radius graph and store, for each connected component k, its centroid $\bar{\mu}_{ik}$, axis-aligned bounding box B_{ik}, and original-support mass $w_{ik} = |\mathcal{P}_{ik}|/|\mathcal{P}_i|$. This is a geometric proxy only: one noun still induces one inventory node, one mask channel, and one rendered support. Thus a node may cover multiple disconnected same-description instances without replacing them by an empty-space-spanning union box. Small isolated components may be ignored only within the uncovered-mass budget, are never merged into a retained component, and retain their original mass in coverage accounting; if the number of retained components exceeds a fixed safety limit, relations involving that node are withheld. All inventory nodes form $\mathcal{V}$.

Pairwise geometric relations. Relations are first scored between component boxes. Let c and d be components of nodes u and v, and let $\mathrm{Vol}(\cdot)$ denote bounding-box volume. Their containment score is

$$q_{\text{in}}(c,d) = \lambda_1 \cdot \frac{\mathrm{Vol}(B_c \cap B_d)}{\mathrm{Vol}(B_c)} + \lambda_2 \cdot \kappa\left(\frac{\mathrm{Vol}(B_d)}{\mathrm{Vol}(B_c)}\right), \tag{5}$$

where $\kappa(x) = \min\big(1,\ \log(1+x)/\log 9\big)$. Here c is the inner candidate, d the container, $\lambda_1 = 0.75$, and $\lambda_2 = 0.25$; the score is zero unless $\mathrm{Vol}(B_c \cap B_d)/\mathrm{Vol}(B_c) \geq 0.85$ and $\mathrm{Vol}(B_d)/\mathrm{Vol}(B_c) \geq 1.8$. Along the vertical axis, define the normalized footprint overlap $\omega_{xy}(c,d) = A_{xy}(B_c \cap B_d)/\min(A_{xy}(B_c), A_{xy}(B_d))$, the vertical gap $\delta_h(c,d) = z_{\min}(B_c) - z_{\max}(B_d)$, and $\delta_h^+ = \max(0, \delta_h)$. The component support score is

$$q_{\text{on}}(c,d) = 0.6\,\omega_{xy}(c,d) + 0.4\left(1 - \frac{\delta_h^+(c,d)}{\tau_{\text{gap}}}\right), \tag{6}$$

and is used only if $\omega_{xy} \geq 0.18$ and $-0.08 \leq \delta_h \leq \tau_{\text{gap}} = 0.30$; otherwise it is zero, apart from a strict center fallback capped at 0.55. For relation r, each source component selects its best target-component witness. We emit a node-level edge only when the passing source mass $\sum_k w_{uk}\mathbb{I}[\max_l q_r(c_{uk}, c_{vl}) \geq \tau_r]$ reaches 0.75; containment assigns this coverage to the contained support. Part edges additionally require lexical whole–part consistency. Kruskal sparsifies undirected candidate pairs; direction and relation type are assigned afterward by bidirectional relation-specific scoring (Appendix B.3). The final graph also contains proximity and directional edges and is therefore not itself a tree.

The default implementation uses these component-aware proxies. In the Union-AABB ablation, all components of a node are replaced by one box enclosing its complete Gaussian support; the inventory node, mask channel, lifted support, and routing configuration remain unchanged.

This yields a weighted graph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$. The visual parser extracts a landmark phrase $\ell(s)$ from the query's target–relation–landmark tuple, and phrase binding maps it to an inventory anchor $a = b(\ell(s))$. For a resolved anchor, its one-hop neighborhood

$$\mathcal{N}(a) = \{u : (u,a) \in \mathcal{E} \text{ or } (a,u) \in \mathcal{E}\} \tag{7}$$

is used to construct subsequent spatial hints. If no landmark binds uniquely, $h(s) = \varnothing$ and routing falls back to full-inventory selection rather than inventing an anchor.

Vision-language routing. Given query s, the vision-language model first reads the query frame, or a small set of keyframes when no frame is bound to the query, and extracts a content phrase $\tilde{o}$ together with the presence of any spatial predicate. Every query undergoes this image-conditioned step. If the visual result uniquely matches an inventory node and contains no spatial predicate, that node is selected directly as v^*, without an inventory-selection call. Otherwise, the original instruction and $\tilde{o}$ are passed to bounded inventory selection using the same model; when a spatial predicate is present, we additionally supply a one-hop neighborhood summary $h(s)$ around the anchor:

$$v^*(s) = \begin{cases} \hat{v}, & \text{direct visual–inventory match,} \\ \Psi(s, \tilde{o}, \mathcal{V}, h(s)), & \text{otherwise.} \end{cases} \tag{8}$$

Inventory selection is text-only over $\mathcal{V}$, with $h(s)$ supplying local context. Routing returns one node, whose support may join multiple same-name instances but cannot compose heterogeneous nodes. In 3D indoor evaluation, a selection that violates support or containment geometry is reselected within $\{v^*\} \cup \mathcal{N}(a)$ or rejected.

Visibility. We output an empty mask if routing returns no node or if the projection of $\mathcal{P}^*$ provides insufficient support within the test camera frustum. This rule gives zero-target and out-of-frustum or insufficient-projection cases a consistent treatment; it does not model physical line-of-sight occlusion.

Table 1: Benchmarks most relevant to hierarchical relations

Dataset	Dim	Hier. Rel.	0/1/G-N	#Sc.	#Query
Ref-LERF	2D	×	×	4	63
Causal-LERF	2D	×	×	4	158
Causal-ScanNet	3D	×	×	10	73
Nest-LERF (Ours)	2D	✓	✓	4	247
Nest-ScanNet (Ours)	3D	✓	✓	10	109

3.4 Rendering and Evaluation

Test-view masks for all inventory nodes are pre-rendered during lifting. Treating support $\mathcal{P}_i$ as foreground, standard 3DGS alpha compositing (Kerbl et al., 2023; Shen et al., 2024) gives

$$\hat{\mathbf{M}}_i = \mathbb{I}\left[\sum_j \alpha_j \prod_{k<j}(1-\alpha_k)\, c_i^{(j)} > \tau_\alpha\right], \qquad c_i^{(j)} = \mathbb{I}[g_j \in \mathcal{P}_i], \tag{9}$$

At query time, 2D evaluation performs no additional rendering: it directly retrieves the pre-rendered mask of selected node v^* and computes pixel IoU against ground truth. If the visibility rule in Section 3.3 deems the node invisible in the test view, an empty mask is evaluated instead. For 3D evaluation, $\mathcal{P}^*$ is compared directly with point-level instance ground truth. Because ScanNet Gaussian centers preserve the original mesh-vertex ordering and coordinates, predicted Gaussian supports and point-level instance annotations lie in the same discrete domain (Appendix Section A.4). Empty targets follow the generalized-referring convention (Liu et al., 2023): an empty prediction with empty ground truth receives full credit, whereas a nonempty prediction for empty ground truth receives zero. We average over queries within each scene and then macro-average across scenes.

4 Experiments

4.1 Benchmarks

We construct Nest-LERF and Nest-ScanNet to evaluate hierarchical relational segmentation. Table 1 presents the most relevant benchmarks along three core dimensions; Appendix Table A.3 provides a complete comparison of sources, task types, and capabilities. Ref-LERF was introduced with ReferSplat (He et al., 2025), while Causal-LERF and Causal-ScanNet originate from CausalSplat (Ding et al., 2026). Nest uses the same LERF and ScanNet scenes as CausalSplat, retaining referring and spatial constraints while adding object-level hierarchical relations and zero-target, single-target, and grouped-multiple-target (0/1/grouped-N) settings. Nest-ScanNet evaluates instances rather than categories.

Nest-LERF contains 247 queries across four LERF (Kerr et al., 2023) scenes, while Nest-ScanNet contains 109 queries across 10 indoor scenes. They use pixel-level and point-level instance ground truth, respectively. Appendix A.3 details the scenes, annotation procedure, and verification protocol.

Hier. Rel. indicates that inter-object hierarchical relations are an explicit evaluation axis; 0/1/G-N denotes zero-target, single-target, and grouped-multiple-target settings. Nest-ScanNet uses instance-level 3D ground truth, whereas Causal-ScanNet follows a category-level protocol.

4.2 Implementation Details

All experiments are implemented in PyTorch and run on a single NVIDIA RTX 4090D. We use SAM 3 (Carion et al., 2026) for 2D segmentation and Qwen3-VL-plus (Bai et al., 2025) for scene-noun proposal, query-image interpretation, and inventory selection. We attach no learnable semantic features to the Gaussians. For LERF, nouns are proposed only from training RGB frames after excluding test views. ScanNet uses all observations because it has no separate test split. Neither setting reads evaluation queries, preserving $\mathcal{O} \perp \mathcal{S}$.

Relations use connected-component centroids and local axis-aligned bounding boxes; graph construction invokes no language model, and the Union-AABB ablation changes only these geometric

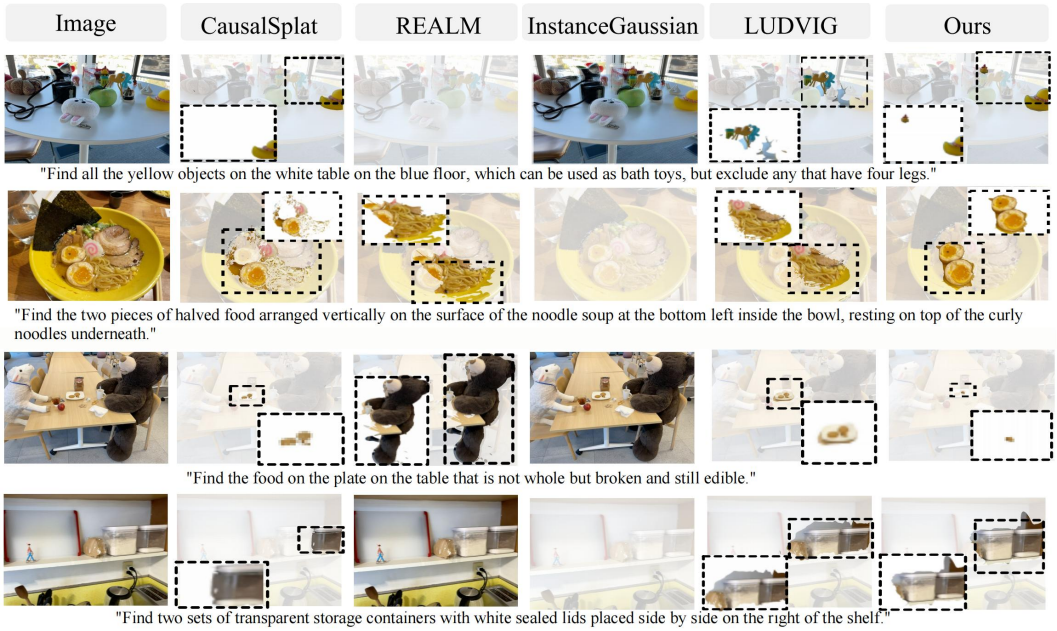


Figure 3: Qualitative comparison of query segmentation on Nest-LERF. Each row preserves the original query, and columns show predictions from the corresponding methods.

proxies. Each query receives visual parsing; a second one-hop inventory-selection call is used only without a unique direct match or when the instruction contains a spatial predicate. Appendices B–C give model, reconstruction, lifting, and threshold details.

4.3 Hierarchical Relational Reasoning Segmentation

Setup. (1) *Task.* Given a natural-language instruction containing nesting, part-based, mixed, and either stacking (Nest-LERF) or spatial-directional (Nest-ScanNet) constraints, the model must localize and segment the target in a 3D Gaussian scene. Unlike standard referring segmentation, which assumes exactly one target, an instruction may denote no target, one target, or a grouped set of same-description targets represented by one inventory support. (2) *Datasets.* Nest-LERF measures 2D understanding by pixel IoU in test views; Nest-ScanNet measures 3D understanding by instance-level point IoU. (3) *Baselines.* We compare three method families: open-vocabulary methods OpenGaussian (Wu et al., 2024), InstanceGaussian (Li et al., 2025), DrSplat (Kim et al., 2025), and LUDVIG (Marrie et al., 2025); referring segmentation method ReferSplat (He et al., 2025); and reasoning methods REALM (Shi et al., 2026) and CausalSplat (Ding et al., 2026). Hyperparameters follow the original papers. Because of representational limitations, pixel-based methods are not evaluated on Nest-ScanNet. (4) *Metric.* We report scene-macro-averaged mIoU: 2D pixel IoU for Nest-LERF and 3D instance IoU for Nest-ScanNet.

Results on Nest-LERF. As shown in Table 2, NestSplat substantially outperforms all baselines in pixel-level segmentation. Figure 3 provides qualitative comparisons. Open-vocabulary methods such as InstanceGaussian and LUDVIG treat a query primarily as category-name matching and lack disambiguation through containment or support. Reasoning methods such as CausalSplat and REALM are not designed for the hierarchical 0/1/grouped-N constraints evaluated here. By combining bounded inventory selection with one-hop geometric context, NestSplat resolves relational constraints and segments the referred targets. The relation-wise breakdown in Appendix Table A.2 shows that NestSplat leads across all four relation types.

Results on Nest-ScanNet. Table 3 shows that NestSplat achieves the best point-level 3D instance segmentation. In Figure 4, feature-matching baselines over-segment or miss targets, while CausalSplat often fails under containment, directional, part-based, and mixed constraints; NestSplat produces substantially cleaner target regions.

Table 2: 2D mask mIoU on Nest-LERF

Method	figurines	ramen	teatime	waldo	Mean
InstanceGaussian	1.62	4.92	3.87	2.67	3.27
DrSplat	1.95	4.76	4.15	2.77	3.41
REALM	2.53	9.99	9.87	3.16	6.39
OpenGaussian	4.46	13.02	7.22	5.13	7.46
ReferSplat	14.62	4.76	13.02	3.66	9.02
LUDVIG	18.37	18.63	14.60	19.04	17.66
CausalSplat	23.31	28.30	35.37	17.13	26.03
NestSplat	**33.28**	**31.24**	**42.09**	**30.02**	**34.16**

Table 3: 3D instance mIoU on Nest-ScanNet

Method	3D mIoU
OpenGaussian	1.73
InstanceGaussian	1.94
DrSplat	3.27
CausalSplat	4.88
LUDVIG	6.38
NestSplat	**27.42**

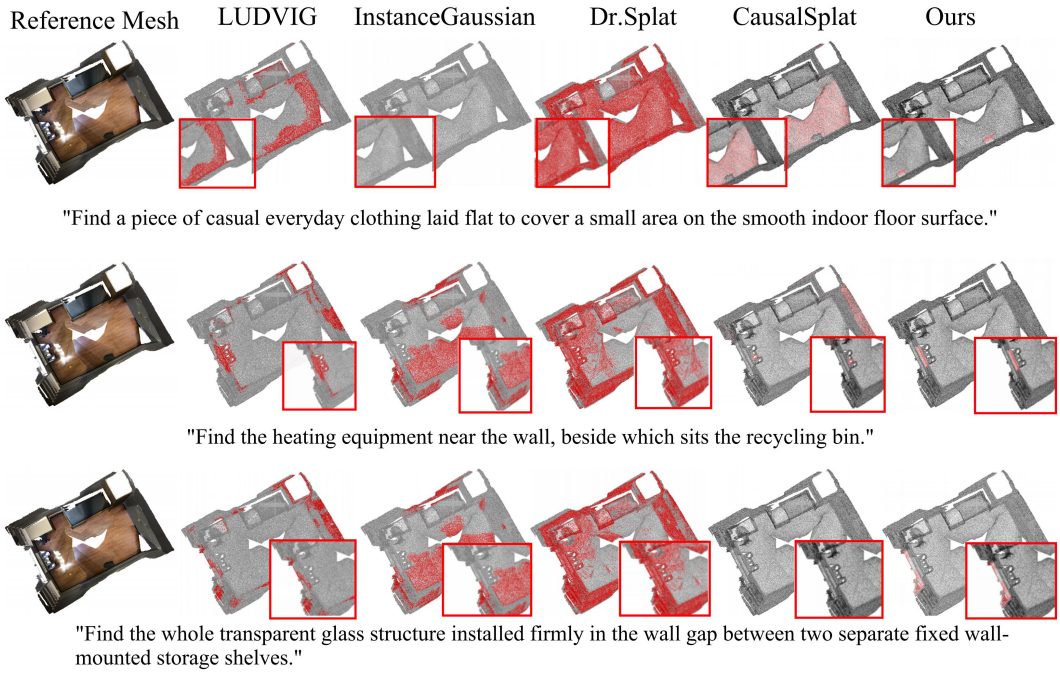


Figure 4: Qualitative comparison of instance segmentation on Nest-ScanNet. The left column shows the reference scene; the remaining columns show predictions from each method.

Table 4: 2D mIoU on Ref-LERF

Method	figurines	ramen	teatime	waldo	Mean
Grounded SAM	16.00	14.10	16.90	16.20	15.80
LangSplat	17.90	12.00	7.60	17.90	13.90
SPIn-NeRF	9.70	7.30	11.70	10.30	9.80
Gaussian Grouping	8.60	27.90	14.80	6.30	14.40
GOI	16.50	27.10	22.90	15.70	20.50
ReferSplat	25.70	35.20	31.30	24.40	29.20
CausalSplat	**50.60**	28.20	38.00	**27.70**	36.10
NestSplat	30.87	**39.98**	**51.21**	26.11	**37.04**

4.4 Transfer to Existing Benchmarks

Without task-specific fine-tuning, NestSplat obtains 37.04% mIoU on Ref-LERF, 48.84% on Causal-LERF, and 27.49% category-level 3D mIoU on Causal-ScanNet. Tables 7–8 summarize task coverage and resource requirements, respectively; Appendix Figures A.3–A.4 provide qualitative comparisons.

NestSplat uses bounded local context with at most two query-time model calls, unlike CausalSplat's graph-dependent multi-hop inference.

Table 5: 2D mIoU on Causal-LERF

Method	figurines	ramen	teatime	waldo	Mean
OpenGaussian	3.30	9.60	4.20	8.20	6.30
DrSplat	16.10	8.70	8.80	13.30	11.70
InstanceGaussian	9.20	2.70	4.70	10.50	6.80
LUDVIG	34.10	32.60	19.70	8.00	23.60
ReferSplat	7.50	5.20	16.20	11.10	10.00
REALM	7.80	9.40	14.30	13.90	11.40
CausalSplat	**46.90**	26.20	68.40	**46.50**	47.00
NestSplat	46.41	**39.12**	**68.43**	41.39	**48.84**

Table 6: 3D category-level mIoU on Causal-ScanNet

Method	3D mIoU
OpenGaussian	2.9
InstanceGaussian	1.4
DrSplat	4
LUDVIG	5.1
CausalSplat	14.9
NestSplat	**27.49**

Table 7: Task coverage comparison

Method	Eval.	Scope	$0/1/G$-N	Opt.	Feat.
ReferSplat	2D	Referring	No	Yes	Yes
REALM	2D	Commonsense	Limited	Yes	Yes
CausalSplat	2D / 3D	Relational	Limited	Yes	Yes
OpenGaussian	3D	Open-vocab.	No	Yes	Yes
LUDVIG	3D	Open-vocab.	No	No	Yes
NestSplat	2D / 3D	Hierarchical	Yes	No	No

G-N: one support may cover multiple same-description instances; heterogeneous multi-node composition is excluded.

Table 8: Efficiency and resource comparison

Method	Scene Prep. (min/scene)	Mean Query Time (s)	Peak Memory (units as reported)
ReferSplat	229.5	0.1	12.2 GB
CausalSplat	525.8	12.8	22.4 GB
LUDVIG	25.7	2.1	10.3 GB
NestSplat	178.7	6.3	17.8 GiB

Table 9: Three-run ablation on Nest-LERF figurines (mean $\pm$ std.)

Variant	mIoU	Δ
Full model (component geometry)	$\mathbf{33.34 \pm 0.29}$	–
Direct Qwen selection (no graph)	31.27 ± 0.41	-2.07
Unstructured proximity graph	31.04 ± 0.03	-2.30
No visual content parsing	32.08 ± 0.75	-1.26
Union-AABB geometry	31.18 ± 1.56	-2.16

4.5 Efficiency and Task Coverage

Table 8 reports 178.7 min scene preparation, 6.3 s mean query latency, and 17.8 GiB peak local-GPU memory. Preparation excludes reconstruction, latency includes network round trips, and memory excludes remote inference. One lifted representation is shared across the three LERF benchmarks; cross-method timings remain approximate because measurement scopes differ.

4.6 Ablation Study

Table 9 compares variants on the same 46 figurines queries with shared masks, inventory, supports, query images, and routing model. No graph, proximity-only, no visual parsing, and Union-AABB reduce three-run mean mIoU by 2.07, 2.30, 1.26, and 2.16 points. Complete diagnostics appear in Appendix Table A.8.

5 Conclusion

We introduce Nest-LERF and Nest-ScanNet to evaluate hierarchical 3DGS segmentation under zero-, single-, and grouped-target settings. NestSplat lifts 2D masks into query-decoupled 3D supports and routes instructions with bounded vision–language decisions and geometric context. It achieves 34.16% scene-macro mIoU on Nest-LERF (8.13 points above CausalSplat) and 27.42% on Nest-ScanNet, with transfer to Ref-LERF and Causal-LERF. Small instances, near-synonymous names, and inventory gaps remain difficult: only 7 of 26 zero-target queries are correctly rejected. Future work will improve absence reasoning and heterogeneous multi-node outputs.

AI Use Statement

Generative AI was used only for language polishing. The authors developed and verified all research ideas, methods, experiments, analyses, and claims and take full responsibility.

References

P. Achlioptas, A. Abdelreheem, F. Xia, M. Elhoseiny, and L. Guibas. ReferIt3D: Neural Listeners for Fine-Grained 3D Object Identification in Real-World Scenes. In *ECCV*, 2020.

S. Bai, Y. Cai, R. Chen, et al. Qwen3-VL Technical Report. arXiv:2511.21631, 2025.

N. Carion, L. Gustafson, Y. T. Hu, et al. SAM 3: Segment Anything with Concepts. In *ICLR*, 2026.

J. Cen, Z. Zhou, J. Fang, C. Yang, W. Shen, L. Xie, D. Jiang, X. Zhang, and Q. Tian. Segment Anything in 3D with NeRFs. In *NeurIPS*, 2023.

J. Cen, J. Fang, C. Yang, L. Xie, X. Zhang, W. Shen, and Q. Tian. Segment Any 3D Gaussians. In *AAAI*, pp. 1971–1979, 2025. doi: 10.1609/aaai.v39i2.32193.

D. Z. Chen, A. X. Chang, and M. Nießner. ScanRefer: 3D Object Localization in RGB-D Scans using Natural Language. In *ECCV*, 2020.

S. Choi, H. Song, J. Kim, T. Kim, and H. Do. Click-Gaussian: Interactive Segmentation to Any 3D Gaussians. In *ECCV*, 2024.

A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *CVPR*, 2017.

J. Ding, M. Song, Y. Chen, W. Gao, and G. Li. CausalSplat: Towards Comprehensive Hierarchical Reasoning in 3D Gaussian Splatting. In *ACM Multimedia*, 2026. Accepted; arXiv:2608.11150.

F. Engelmann, F. Manhardt, M. Niemeyer, K. Tateno, M. Pollefeys, and F. Tombari. OpenNeRF: Open Set 3D Neural Scene Segmentation with Pixel-Wise Features and Rendered Novel Views. In *ICLR*, 2024.

Q. Gu, A. Kuwajerwala, S. Morin, K. M. Jatavallabhula, et al. ConceptGraphs: Open-Vocabulary 3D Scene Graphs for Perception and Planning. In *ICRA*, pp. 5021–5028, 2024. doi: 10.1109/ICRA57147.2024.10610243.

J. Guo, X. Ma, Y. Fan, H. Liu, and Q. Li. Semantic Gaussians: Open-Vocabulary Scene Understanding with 3D Gaussian Splatting. *IEEE Transactions on Circuits and Systems for Video Technology*, 36(7):9265–9277, 2026. doi: 10.1109/TCSVT.2026.3675320.

S. He, G. Jie, C. Wang, Y. Zhou, S. Hu, G. Li, and H. Ding. ReferSplat: Referring Segmentation in 3D Gaussian Splatting. In *ICML*, 2025.

B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao. 2D Gaussian Splatting for Geometrically Accurate Radiance Fields. In *ACM SIGGRAPH*, 2024a.

Z. Huang, X. Wu, X. Chen, H. Zhao, L. Zhu, and J. Lasenby. OpenIns3D: Snap and Lookup for 3D Open-vocabulary Instance Segmentation. In *ECCV*, 2024b.

U. Jain, A. Mirzaei, and I. Gilitschenski. GaussianCut: Interactive Segmentation via Graph Cut for 3D Gaussian Splatting. In *NeurIPS*, 2024.

L. Jiang, S. Shi, and B. Schiele. Open-Vocabulary 3D Semantic Segmentation with Foundation Models. In *CVPR*, pp. 21284–21294, 2024.

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

J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik. LERF: Language Embedded Radiance Fields. In *ICCV*, 2023.

C. M. Kim, M. Wu, J. Kerr, K. Goldberg, M. Tancik, and A. Kanazawa. GARField: Group Anything with Radiance Fields. In *CVPR*, pp. 21530–21539, 2024.

J. S. Kim, G. Kim, Y. J. Kim, Y. C. F. Wang, J. Choe, and T. H. Oh. Dr. Splat: Directly Referring 3D Gaussian Splatting via Direct Language Embedding Registration. In *CVPR*, 2025.

A. Kirillov, E. Mintun, N. Ravi, et al. Segment Anything. In *ICCV*, pp. 4015–4026, 2023.

S. Koch, N. Vaskevicius, M. Colosi, P. Hermosilla, and T. Ropinski. Open3DSG: Open-Vocabulary 3D Scene Graphs from Point Clouds with Queryable Objects and Open-Set Relationships. In *CVPR*, pp. 14183–14193, 2024.

H. Li, Y. Wu, J. Meng, Q. Gao, Z. Zhang, R. Wang, and J. Zhang. InstanceGaussian: Appearance-Semantic Joint Gaussian Representation for 3D Instance-Level Perception. In *CVPR*, 2025.

C. Liu, H. Ding, and X. Jiang. GRES: Generalized Referring Expression Segmentation. In *CVPR*, 2023.

T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai. Scaffold-GS: Structured 3D Gaussians for View-Adaptive Rendering. In *CVPR*, 2024.

J. Marrie, R. Menegaux, M. Arbel, D. Larlus, and J. Mairal. LUDVIG: Learning-Free Uplifting of 2D Visual Features to Gaussian Splatting Scenes. In *ICCV*, 2025.

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

A. Mirzaei, T. Aumentado-Armstrong, K. G. Derpanis, J. Kelly, M. A. Brubaker, I. Gilitschenski, and A. Levinshtein. SPIn-NeRF: Multiview Segmentation and Perceptual Inpainting with Neural Radiance Fields. In *CVPR*, 2023.

P. Nguyen, T. D. Ngo, E. Kalogerakis, C. Gan, A. Tran, C. Pham, and K. Nguyen. Open3DIS: Open-Vocabulary 3D Instance Segmentation with 2D Mask Guidance. In *CVPR*, pp. 4018–4028, 2024.

S. Peng, K. Genova, C. Jiang, A. Tagliasacchi, M. Pollefeys, and T. Funkhouser. OpenScene: 3D Scene Understanding with Open Vocabularies. In *CVPR*, pp. 815–824, 2023.

M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister. LangSplat: 3D Language Gaussian Splatting. In *CVPR*, 2024.

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

A. Radford, J. W. Kim, C. Hallacy, et al. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*, 2021.

T. Ren, S. Liu, A. Zeng, et al. Grounded SAM: Assembling Open-World Models for Diverse Visual Tasks. arXiv:2401.14159, 2024.

D. Rozenberszki, O. Litany, and A. Dai. UnScene3D: Unsupervised 3D Instance Segmentation for Indoor Scenes. In *CVPR*, pp. 19957–19967, 2024.

J. L. Schönberger and J. M. Frahm. Structure-from-Motion Revisited. In *CVPR*, 2016.

Qiuhong Shen, Xingyi Yang, and Xinchao Wang. Flashsplat: 2d to 3d gaussian splatting segmentation solved optimally. In *European Conference on Computer Vision*, 2024.

C. Shi, M. Chen, Y. Mao, C. Yang, X. Hu, J. Ding, and Z. Yu. REALM: An MLLM-Agent Framework for Open World 3D Reasoning Segmentation and Editing on Gaussian Splatting. In *CVPR*, 2026.

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

A. Takmaz, A. Delitzas, R. W. Sumner, F. Engelmann, J. Wald, and F. Tombari. Search3D: Hierarchical Open-Vocabulary 3D Segmentation. *IEEE Robotics and Automation Letters*, 2025.

J. Wald, H. Dhamo, N. Navab, and F. Tombari. Learning 3D Semantic Scene Graphs From 3D Indoor Reconstructions. In *CVPR*, pp. 3961–3970, 2020.

Z. Wei, J. Lin, Y. Liu, W. Chen, J. Luo, G. Li, and L. Lin. 3DAffordSplat: Efficient Affordance Reasoning with 3D Gaussians. In *ACM Multimedia*, 2025.

Y. Wu, J. Meng, H. Li, C. Wu, Y. Shi, X. Cheng, C. Zhao, H. Feng, E. Ding, J. Wang, and J. Zhang. OpenGaussian: Towards Point-Level 3D Gaussian-based Open Vocabulary Understanding. In *NeurIPS*, 2024.

Y. Xie, A. Arafa, A. Javanmardi, C. Millerdurai, J. C. Hu, S. Wang, A. Pagani, and D. Stricker. ReLaGS: Relational Language Gaussian Splatting. In *CVPR*, pp. 23826–23836, 2026.

M. Ye, M. Danelljan, F. Yu, and L. Ke. Gaussian Grouping: Segment and Edit Anything in 3D Scenes. In *ECCV*, 2024.

Y. Yin, Y. Liu, Y. Xiao, D. Cohen-Or, J. Huang, and B. Chen. SAI3D: Segment Any Instance in 3D Scenes. In *CVPR*, pp. 3292–3302, 2024.

H. Ying, Y. Yin, J. Zhang, F. Wang, T. Yu, R. Huang, and L. Fang. OmniSeg3D: Omniversal 3D Segmentation via Hierarchical Contrastive Learning. In *CVPR*, pp. 20612–20622, 2024.

Z. Yu, A. Chen, B. Huang, T. Sattler, and A. Geiger. Mip-Splatting: Alias-free 3D Gaussian Splatting. In *CVPR*, 2024.

C. Zhang, A. Delitzas, F. Wang, R. Zhang, X. Ji, M. Pollefeys, and F. Engelmann. Open-Vocabulary Functional 3D Scene Graphs for Real-World Indoor Spaces. In *CVPR*, pp. 19401–19413, 2025.

Y. Zhang, Z. Gong, and A. X. Chang. Multi3DRefer: Grounding Text Description to Multiple 3D Objects. In *ICCV*, pp. 15225–15236, 2023.

S. Zhou, H. Chang, S. Jiang, Z. Fan, Z. Zhu, D. Xu, P. Chari, S. You, Z. Wang, and A. Kadambi. Feature 3DGS: Supercharging 3D Gaussian Splatting to Enable Distilled Feature Fields. In *CVPR*, 2024.

Appendix

This appendix describes benchmark construction, implementation and evaluation details, additional qualitative results, failure analysis, and downstream scene editing.

Additional related work. Point-cloud segmentation includes open-vocabulary semantics and instances (Jiang et al., 2024; Takmaz et al., 2023; Huang et al., 2024b; Nguyen et al., 2024; Yin et al., 2024; Rozenberszki et al., 2024); Search3D (Takmaz et al., 2025) extends open-vocabulary search to parts. GARField and OmniSeg3D study hierarchical grouping in radiance fields (Kim et al., 2024; Ying et al., 2024). ScanRefer and ReferIt3D study single-object 3D grounding (Chen et al., 2020; Achlioptas et al., 2020); 3DSSG and OpenFunGraph model spatial or functional scene relations (Wald et al., 2020; Zhang et al., 2025). These settings complement our relational, mask-level benchmark.

A Benchmark Construction and Evaluation Protocol

A.1 Relation Taxonomy and Dataset Differences

Nest-LERF and Nest-ScanNet evaluate language-guided segmentation in 3D Gaussian scenes. Queries describe targets through object relations and attributes and may denote no target, one physical target, or a grouped set of same-description physical targets represented by one noun-anchored support. Cardinality is defined by the ground-truth instances merged into the target mask, independently of the model-side inventory-node count. Each dataset uses a taxonomy tailored to its scene structure. Table A.1 defines the Nest-LERF categories; Nest-ScanNet uses nesting, part-based, spatial-directional, and mixed-relation queries, defined below.

Table A.1: Relation categories and decision criteria in Nest-LERF

Type	Definition	Decision Criterion	Example Query
Nesting	Most of a small object's volume lies inside a larger container	The target is described through container containment	"Find the objects stacked in rows inside the transparent box behind the white table"
Stacking	Vertical support order: the upper bottom face approaches the lower top face, with overlapping projections	Emphasizes vertical contact and support order	"Find the silver object beneath the plush rabbit's ears on the white table"
Part-based	The target is identified through its relation to a component or attachment	A part may identify the whole, or the query may target a part or attachment; the referred entity defines the annotated extent	"Find the transparent electric kettle with a metal top, used to boil drinking water, below the shelf"
Mixed Relations	One instruction combines several relation types	All relational constraints in the instruction must be satisfied jointly	"On the white table, find the green object between the black camera and the red chair"

Table A.2: Query-level 2D mask mIoU (%) on Nest-LERF by relation type

Method	Nesting (71)	Stacking (127)	Part-based (19)	Mixed (30)
InstanceGaussian	10.69	5.96	0.00	0.00
DrSplat	2.41	4.26	3.08	4.38
REALM	4.72	8.89	5.92	6.16
OpenGaussian	4.98	10.56	3.38	8.83
ReferSplat	2.81	12.55	3.96	8.54
LUDVIG	14.11	18.17	14.75	25.04
CausalSplat	<u>17.55</u>	<u>28.97</u>	<u>33.13</u>	<u>36.63</u>
NestSplat	**26.09**	**38.18**	**34.58**	**37.15**

All values are arithmetic means over the queries in each relation type; query counts appear in parentheses. The NestSplat row uses the updated raw-precision IoUs from the N-column aggregation.

Table A.3: Comprehensive comparison of 3D Gaussian understanding and reasoning benchmarks

Dataset	Source	Task	Dim	Domain	#Sc.	#Query
LERF	LangSplat	Open-Vocab Seg.	2D	Real	4	208 Words
ScanNet	OpenGaussian	Open-Vocab Seg.	3D	Real	10	84 Words
Ref-LERF	ReferSplat	Ref. Seg.	2D	Real	4	63 Inst.
3DAffordSplat	3DAffordSplat	Reason. Seg.	3D	Synth.	8.3K	6.6K Inst.
REALM3D	REALM	Reason. Seg.	2D	Real	100+	1K+ Pairs
Causal-LERF	CausalSplat	Reason. Seg.	2D	Real	4	158 Inst.
Causal-ScanNet	CausalSplat	Reason. Seg.	3D	Real	10	73 Inst.
Nest-LERF (Ours)	NestSplat	Hier. Rel. Seg.	2D	Real	4	247 Inst.
Nest-ScanNet (Ours)	NestSplat	Hier. Rel. Seg.	3D	Real	10	109 Inst.

Dataset	Ref.	Spa.	Imp. Reas.	Hier. Rel.	0/1/G-N
LERF	×	×	×	×	×
ScanNet	×	×	×	×	×
Ref-LERF	✓	✓	×	×	×
3DAffordSplat	×	×	✓	×	×
REALM3D	✓	✓	✓	×	×
Causal-LERF	✓	✓	✓	×	×
Causal-ScanNet	✓	✓	✓	×	×
Nest-LERF (Ours)	✓	✓	×	✓	✓
Nest-ScanNet (Ours)	✓	✓	×	✓	✓

Nest-ScanNet categories. Nest-ScanNet targets indoor scenes and uses four labels: nesting, part-based relations, spatial-directional relations, and mixed relations.

- **Nesting:** describes a target through containment or embedding, including objects inside containers and objects embedded in walls, countertops, or recesses, such as a sink set into a counter.
- **Part-based relations:** identify a target through its relation to a component, attachment, or functionally associated object. Nest-LERF emphasizes whole–part and attachment relations, while Nest-ScanNet also includes functional associations, such as locating an item from its role in holding tissues or soap.
- **Spatial-directional relations:** use positional constraints such as left/right, front/behind, above/below, and proximity, e.g., a trash bin below a countertop. Unlike the contact-and-support constraint of Stacking, this category emphasizes relative position.
- **Mixed relations:** combine the preceding relations or functional cues. Relation type and target cardinality are annotated separately.

Section A.4 reports the category distribution under these labels, and Figure A.1 illustrates the four Nest-LERF query types.

Query-writing principles. Queries describe targets using object relations, appearance attributes, and functional cues. When a landmark is involved, the relation between target and landmark is stated explicitly. Target extent is defined by an instance annotation or pixel mask. The sources represented in Appendix Table A.3 are LangSplat (Qin et al., 2024), OpenGaussian (Wu et al., 2024), ReferSplat (He et al., 2025), 3DAffordSplat (Wei et al., 2025), REALM (Shi et al., 2026), and CausalSplat (Ding et al., 2026).

Abbreviations: Seg., segmentation; Ref., referring; Inst., instruction. Imp. Reas. combines commonsense, affordance, and counterfactual reasoning. Hier. Rel. denotes benchmarks that explicitly evaluate inter-object hierarchical relations; 0/1/G-N indicates zero-target, single-target, and grouped-multiple-target queries. Nest-ScanNet uses instance-level 3D ground truth, whereas Causal-ScanNet follows a category-level protocol.

A.2 Scenes and Scale

Nest uses the same LERF (Kerr et al., 2023) and ScanNet (Dai et al., 2017) scenes as CausalSplat (Ding et al., 2026), with newly written instructions and independent annotations.

Nest-LERF. The `figurines`, `ramen`, `teatime`, and `waldo_kitchen` scenes contain 46, 82, 67, and 52 query samples, respectively, for a total of 247. Each sample consists of an instruction and a test view bound by frame identifier, with pixel IoU computed in that view. The dataset contains 186 distinct instruction strings; the same text may be bound to multiple test views, each routed and scored independently. Query counts refer to instruction–view pairs.

Nest-ScanNet. We use the ten indoor scenes adopted by OpenGaussian (Wu et al., 2024), with 109 queries in total and ScanNet instance IDs as 3D ground truth. The scenes are:

`scene0000_00`, `scene0062_00`, `scene0070_00`, `scene0097_00`, `scene0140_00`, `scene0200_00`, `scene0347_00`, `scene0400_00`, `scene0590_00`, and `scene0645_00`.

Causal-ScanNet uses category-level ground truth, while Nest-ScanNet uses instance identifiers. They evaluate category localization and instance discrimination, respectively (Tables 6 and 3).

A.3 Annotation Procedure

All instructions are manually annotated and cross-checked in three steps.

(1) Instruction writing. Annotators inspect LERF test frames and ScanNet RGB sequences with instance-colored point clouds, then write English queries according to the categories in Section A.1. For a zero-target instruction, no scene instance satisfies the described conditions.

(2) Ground-truth annotation.

- Nest-LERF: annotate a pixel mask for the target in the specified test view. For grouped-multiple targets, all referred same-description instances represented by one support are merged into one target mask; zero-target masks are empty.
- Nest-ScanNet: verify instance IDs in `aggregation.json` and record one or more `gt_instance_ids`. Grouped-multiple-target ground truth is the union of the referred same-description instance point sets; a zero target corresponds to an empty point set.

(3) Cross-checking. A second annotator independently verifies the validity of instance IDs, the geometric direction of containment and support relations, zero-target decisions, and the completeness of grouped-multiple-target annotations. Disagreements are adjudicated by a third annotator before inclusion.

Query and ground-truth files record the scene, instruction, and target instance or pixel mask; the source taxonomy table separately provides relation-type and cardinality labels. Scene preparation reads neither evaluation queries nor their annotations. Query routing uses only the instruction text, scene images, and the precomputed inventory and geometric relations, never ground-truth fields. We use $\mathcal{O} \perp \mathcal{S}$ to denote this procedural decoupling: object nouns are generated from scene observations, and evaluation queries enter only during query inference.

A.4 Cardinality Distribution and Scoring

Table A.4 reports the per-scene query-cardinality distribution of Nest-LERF together with mIoU by target cardinality. Among 247 queries, 26 (10.5%) have zero targets, 151 (61.1%) have one target, and 70 (28.3%) have grouped-multiple targets.

Under the manual labels, the 109 Nest-ScanNet queries comprise 11 zero-target (10.1%), 60 single-target (55.0%), and 38 grouped-multiple-target (34.9%) cases. The unit is a scene–instruction query sample, so the same instruction occurring in different scenes is counted separately. Under the original relation labels, nesting, part-based, spatial-directional, and mixed relations account for 24 (22.0%), 31 (28.4%), 40 (36.7%), and 14 (12.8%) queries, respectively.

Scoring follows Section 3.4:

Table A.4: Nest-LERF query counts and mIoU by target cardinality

(a) Query counts

Scene	Zero	Single	G-N	Total
figurines	4	36	6	46
ramen	5	43	34	82
teatime	12	38	17	67
waldo_kitchen	5	34	13	52
Total	26	151	70	247

(b) Query-level mIoU

Model	Zero	Single	G-N	Overall
REALM	0	0.0810	0.0770	0.0713
DrSplat	0	0.0367	0.0497	0.0365
InstanceGaussian	0.0385	0.0321	0.0418	0.0355
LUDVIG	0	0.2170	0.1519	0.1757
OpenGaussian	0	0.0871	0.1011	0.0819
ReferSplat	0	0.0894	0.1108	0.0861
CausalSplat	0	0.2729	<u>0.3617</u>	0.2694
NestSplat	**0.2692**	**0.3418**	**0.3732**	**0.3431**

- **Single / grouped-multiple targets:** compute IoU between the predicted and ground-truth masks. For a grouped-multiple-target query, ground-truth masks of all referred same-description instances are merged before computing IoU.
- **Zero targets:** score 1.0 when both prediction and ground truth are empty, and 0.0 otherwise.
- **Averaging:** Nest-LERF scores each instruction–test-view pair as one query sample, computing IoU independently between its predicted and ground-truth masks. If one instruction is bound to two test views, it forms two samples, each with its own IoU and contribution to the average. A scene score is the sum of sample IoUs divided by the number of samples in that scene. Table 2 takes the arithmetic mean of the four scene scores, yielding a scene macro-average of 34.16%. Table A.4 averages the IoUs of all 247 samples, yielding a query-level average of 34.31%. The former weights scenes equally; the latter weights query samples equally. Because scenes contain 46–82 samples, the two schemes produce slightly different aggregate scores.

Nest-ScanNet's 3D evaluation directly compares selected-instance support $\mathcal{P}^*$ with point-level instance ground truth. Empty targets use the same binary scoring rule.

For ScanNet evaluation, the Gaussian representation preserves the original ScanNet vertex topology. Densification and pruning are disabled, and the learning rate of Gaussian centers is set to zero. Across all ten scenes, the 90k Gaussian centers, ScanNet mesh vertices, and `segIndices` contain 1,569,975 entries with identical ordering. Index-wise coordinate distances have zero median, 95th percentile, and maximum error, establishing a one-to-one correspondence between Gaussian predictions and point-level ground truth. Exact-coordinate matching is 100%, and no scene contains duplicate XYZ coordinates.

Table A.5: Index-preserving correspondence used for Nest-ScanNet evaluation.

Scene	Mesh vertices	90k Gaussians	segIndices
scene0000_00	81,369	81,369	81,369
scene0062_00	51,610	51,610	51,610
scene0070_00	109,380	109,380	109,380
scene0097_00	72,007	72,007	72,007
scene0140_00	372,941	372,941	372,941
scene0200_00	83,291	83,291	83,291
scene0347_00	67,984	67,984	67,984
scene0400_00	155,959	155,959	155,959
scene0590_00	222,957	222,957	222,957
scene0645_00	352,477	352,477	352,477
Total	**1,569,975**	**1,569,975**	**1,569,975**

A.5 Visibility and Empty Masks

2D evaluation retrieves pre-rendered masks (Eq. (9)). If routing returns no resolvable target node, the prediction is empty. For a selected node, we take up to 4,096 centers from its Gaussian support, uniformly sampling when the support exceeds this limit, and project them into the test camera. Let n be the number of sampled points and n_{in} the number that project inside the frustum. We set the prediction to an empty mask when $n_{\text{in}} < 32$ and $n_{\text{in}}/n < 2\%$. If no usable 3D points are available, we retain the subsequent mask-retrieval or rendering result.

This check measures frustum coverage through both the projected-point count and its ratio, handling insufficient projection support; it does not model occlusion. Scene-level zero targets are defined by ground-truth annotations, and empty predictions are scored as in Section A.4.

B Implementation Details and Protocols

B.1 Hardware, Models, and Reconstruction

- Compute hardware: NVIDIA RTX 4090D. The environment uses PyTorch 2.3.0 with CUDA 12.1. Key dependencies include OpenAI Python SDK 2.16.0, NumPy 2.2.6, Open3D 0.19.0, NetworkX 3.4.2, and SciPy 1.15.2.
- 2D segmentation model: we use the initial SAM 3 release from Meta in November 2025 (Carion et al., 2026), with official pretrained weights and a local deployment from source. During scene preparation, object-noun phrases generated from scene observations serve as text prompts for producing 2D masks over multi-view RGB images, which are then lifted to 3D Gaussian supports. Query-time routing operates on the precomputed scene representation and triggers no additional SAM 3 segmentation.
- Vision-language model: temperature 0.1; the same model is used for scene object-noun generation, visual query interpretation, and inventory selection.
- Gaussians: no learnable semantic features are attached to $\mathcal{G}$. LERF reconstructions are trained for 30k iterations and ScanNet reconstructions for 90k. Scene-preparation time is measured after reconstruction is complete.

B.2 Object-Noun Protocol $\mathcal{O} \perp \mathcal{S}$

The nouns used for lifting are generated from scene observations without access to the evaluation-query set.

- LERF: exclude test views and propose nouns only from training RGB frames.
- ScanNet: use all RGB frames for noun proposal.
- Images are sent to the vision-language model in batches of at most 48 frames. Across the four Nest-LERF scenes, the total number of batches is $\sum \lceil N_{\text{train}}/48 \rceil = 18$.
- A subsequent text-only call merges synonymous nouns; each scene inventory contains at most 96 entries.
- Multi-view mask filtering: for each object-noun phrase, we count the views producing a valid segmentation mask and retain only phrases meeting a minimum-view requirement. The minimum is 4 for Ref-LERF and Causal-LERF, 5 for the figurines-specific configuration, and 2 for Nest-ScanNet.
- Contribution-based lifting follows Eqs. (1)–(2); foreground Gaussians with conflict ratio $\rho_g > \tau_{\text{bg}} = 0.8$ are relabeled as background.

After scene preparation, both visual query interpretation and inventory selection operate on the fixed lifted representation.

B.3 Geometric-Relation Hyperparameters

Table A.6 summarizes the parameters reported in Section 4.2. Shared geometric parameters remain fixed across scenes; dataset-specific checks and lifting filters are detailed in Sections B.4 and B.2, respectively.

Relation edges are computed from connected-component centroids and axis-aligned boxes, without language-model calls during graph construction. The radius is three times a robust nearest-neighbor spacing estimated by querying the full support; component membership is then computed on all support points. Components below 64 points are discarded only when their total mass is at most $1 - 0.75$, without reassigning their points or renormalizing retained mass. More than 32 retained components disables geometric relations for that node rather than recreating a union box. Candidate generation and relation scoring use component pairs; a node-level relation requires at least 0.75 source-mass coverage. For each unordered node pair, the maximum local component-overlap or proximity score defines an undirected affinity, on which Kruskal builds a maximum spanning forest.

Table A.6: Parameters for geometric relations and view checks

Symbol	Meaning	Value
λ_1, λ_2	Containment: volume overlap / scale ratio	0.75 / 0.25
λ_3, λ_4	Support: horizontal overlap / vertical gap	0.6 / 0.4
τ_{gap}	Support-gap threshold	0.30 m
Shared	Minimum horizontal overlap	0.18
Shared	Proximity centroid distance	2.5 m
Shared	Edge-score pruning threshold	0.45
Components	Min. points / max. proxies / coverage	64 / 32 / 0.75
τ_{bg}	Cross-view conflict ratio	0.8
τ_{α}	Pre-rendered foreground threshold	0.5
Shared	In-frustum point count / sampled-point ratio	32 / 2%
Shared	Proximity threshold for the Nest-ScanNet bounding-box consistency check	1.75 m

For every retained pair, support, containment, and part–whole scores are then evaluated in both directions with priority `on` $>$ `contains` $>$ `part_of`; the selected family emits directed `on/under` or `contains/inside` inverse edges, or one directed `part_of` edge. Thus Kruskal selects candidate pairs but does not erase direction or relation type. The emitted audit records component counts, ignored mass, reliability, and union-to-component inflation. The final graph also contains organizational, proximity, and directional edges and is therefore not itself a tree.

B.4 Query-Time Routing

Each instruction is first passed to the vision-language model together with its bound test frame. When a ScanNet query has no bound frame, three frames are sampled uniformly from the RGB sequence. The model returns fields including a target phrase, a candidate inventory match, and spatial cues. We select the candidate node directly when the match resolves to an inventory node, neither the model nor text rules detect a spatial cue, and the reported confidence is at least 0.55. The direct-selection branch also permits responses that omit the confidence field. All other cases invoke one text-only inventory-selection call, combining the original instruction with the visually inferred target phrase, candidate match, and landmark information, plus a one-hop neighborhood summary $h(s)$ when available. If inventory selection returns no valid node but the earlier stage produced a resolvable match, the implementation falls back to that match.

Inventory entries contain a node identifier, lifted name, and aliases. Routing selects at most one node, whose Gaussian support may cover one physical instance or multiple disconnected physical instances sharing the same semantic description; their joint projection into the test view forms the predicted mask. Accordingly, target cardinality is measured in physical-instance space and need not equal the number of selected nodes.

The bounding-box geometric-consistency check is enabled only for Nest-ScanNet; the reselection step is disabled for Causal-LERF and Causal-ScanNet. Routing returns a `target_node`; empty outputs arise from node parsing and the frustum checks in Section A.5.

B.5 Random Sampling and Caching

Random sampling. When target support exceeds 4,096 Gaussian centers, the frustum check samples 4,096 centers uniformly without replacement. This step has no dedicated random seed, so its output depends on the process state at run time. Parts of the lifting and rendering pipeline initialize the Python, NumPy, and PyTorch random seeds to 0; sampling during query evaluation inherits the random state of its process. **Caching.** A cached scene object-noun list is reused when valid unless rebuilding is requested. During 2D evaluation with visual-content routing, routing results for identical images and instructions are reused within a run, but response caches are not shared across runs. Per-sample predictions are saved separately.

C Hyperparameters and Module Analysis

C.1 Controlled Update of the Strongest Baseline

To test whether the Nest-LERF gain is explained only by the query-time vision–language model, we rerun CausalSplat, the strongest baseline in Table 2, after replacing its original query-time model with Qwen3-VL-plus. We retain CausalSplat's original SAM-v1 mask-generation and scene-construction pipeline, so this experiment changes only the query reasoner. Table A.7 shows that the update improves CausalSplat from 26.03% to 28.35% mean mIoU, while NestSplat remains 5.81 percentage points higher.

Table A.7: Controlled update of the strongest Nest-LERF baseline

Method	figurines	ramen	teatime	waldo	Mean
CausalSplat (original)	23.31	28.30	35.37	17.13	26.03
CausalSplat (updated VLM)	30.22	28.43	35.37	19.37	28.35
NestSplat	**33.28**	**31.24**	**42.09**	**30.02**	**34.16**

C.2 Component Analysis

Table 9 reports five figurines results directly related to the core design; Table A.8 gives the complete ablation, including secondary components.

Shared-front-end controls. The first five variants in Tables 9 and A.8 use the same SAM 3 masks, routing model, 94-node inventory, lifted Gaussian supports, and query images, and report mean $\pm$ standard deviation over three runs. The direct-selection variant preserves visual query interpretation and selection over the complete inventory but removes the graph-derived summary $h(s)$. The proximity-only variant replaces typed containment, support, part, and directional context with unstructured proximity edges. The remaining three rows are single-run secondary diagnostics whose deltas use the matched full-model score of 33.28%.

Shared geometric parameters. The shared parameters in Table A.6 are used for all four Nest-LERF scenes and all ten Nest-ScanNet scenes, without per-scene search. Across the repeated figurines evaluation, replacing component-aware proxies with a single Union AABB decreases mean mIoU by 2.16 points and increases the run-to-run standard deviation from 0.29 to 1.56. Replacing containment, support, and part relations with an unstructured proximity graph reduces mean mIoU by 2.30 points, the largest repeated mean drop. The single-run edge-pruning diagnostic decreases mIoU by 0.10 points.

Scope of the geometric-consistency check. Enabling the Nest-ScanNet bounding-box conflict reselection on figurines reduces mIoU to 31.44%, 1.84 percentage points below the default. The degradation may stem from depth noise in tabletop scenes: geometric errors can make support and containment constraints overly strict and exclude the correct instance. By default, the check is disabled on Nest-LERF and enabled on Nest-ScanNet; it is also disabled for transfer evaluation on Causal-LERF and Causal-ScanNet.

Floor–room labels. Removing these labels on figurines decreases mIoU by 2.05 percentage points. In tabletop scenes, the labels provide coarse grouping. Consistent with Section 3.3, ScanNet retains the floor–room structure to organize indoor scenes, although the current benchmark contains few cross-room instructions.

Table A.8: Complete ablation on Nest-LERF figurines. Core controls report mean ± std. over three runs; secondary diagnostics are single-run.

Variant	mIoU	Δ
Repeated core controls (3 runs)		
Full model (component geometry)	**33.34 ± 0.29**	–
Direct Qwen selection (no graph)	31.27 ± 0.41	-2.07
Unstructured proximity graph	31.04 ± 0.03	-2.30
No visual content parsing	32.08 ± 0.75	-1.26
Union-AABB geometry	31.18 ± 1.56	-2.16
Single-run secondary diagnostics		
No organizational labels	31.23	-2.05
No edge pruning	33.18	-0.10
Additional geometric-consistency check	31.44	-1.84

C.3 Component-Geometry Audit

We audit the geometric effect of replacing Union AABBs with component-aware proxies on the frozen Nest-LERF figurines scene. Both branches share the same 94 inventory nodes, SAM 3 masks, lifted Gaussian supports, query images, Qwen3-VL-plus model, and routing parameters; only relation construction changes. Components are geometric proxies only: they create neither additional selectable nodes nor additional mask channels.

Table A.9: Geometry audit on the frozen Nest-LERF figurines scene. Inflation is the Union-AABB value divided by its component-aware counterpart.

Metric	Value
Inventory nodes	94
Multi-component nodes	89
Unreliable nodes	3
Volume inflation, median / P95 / max	$3.62\times$ / 2.82×10^6 / 7.24×10^6
Diagonal inflation, median / P95 / max	$1.56\times$ / $144\times$ / $353\times$
Union-AABB edges → component-aware edges	$867 \rightarrow 731$
Removed / added edges	412 / 276
Relation-type changes on the same node pair	81

Component-aware boxes substantially alter the relation graph, removing 412 and adding 276 edges. Among hierarchical containment and support relations, the removed edges include 53 `inside`, 42 `contains`, and 23 `on` edges. The extreme P95 and maximum inflation values arise from a small number of scattered Gaussian outliers, whereas the medians characterize typical nodes. The 89 multi-component nodes nevertheless remain single selectable inventory entries rather than being split into new physical instances.

Table A.10: Routing and repeated-evaluation audit on Nest-LERF figurines (46 queries). mIoU is mean ± std. over three runs.

Metric	Value
Spatial-hint text changed	10
Neighbor set changed	8
Final routed node changed (first run)	4
Final routed node changed (three-run majority)	6
Routing jitter, Union / component	16 / 14
Union-AABB mIoU	$31.18 \pm 1.56\%$
Component-aware mIoU	**33.34 ± 0.29**%

Thus, component-aware boxes substantially change the relation graph, while the final routed node changes for 4 of 46 queries in the first run and 6 queries after three-run majority aggregation. Across the three evaluations, component-aware geometry obtains $33.34 \pm 0.29\%$ mIoU compared with $31.18 \pm 1.56\%$ for Union-AABB, a 2.16-point difference in means. Together with the lower run-

to-run variation, these results show that component-aware proxies provide a more faithful and stable geometric basis for relational routing.

D Additional Qualitative Analysis

Figures 3–4 show representative comparisons on Nest-LERF and Nest-ScanNet. This section adds examples for all four relation types and target cardinalities, as well as the Ref-LERF, Causal-LERF, and Causal-ScanNet results discussed in Section 4.4. Failure cases appear in Figure A.5 of Section E. These examples explain prediction behavior; aggregate performance is given by the main evaluation tables.

D.1 Figure A.1: Qualitative Comparison across Four Relation Types

Figure A.1 presents representative Nest-LERF results for Nesting, Stacking, Part-based, and Mixed Relations queries. The category labels appear in that order on the left. From left to right, the four columns show the original image, ground-truth mask, InstanceGaussian, and our method. Each row retains the original English query and uses the same scene viewpoint, enabling direct comparison between predictions and ground truth.

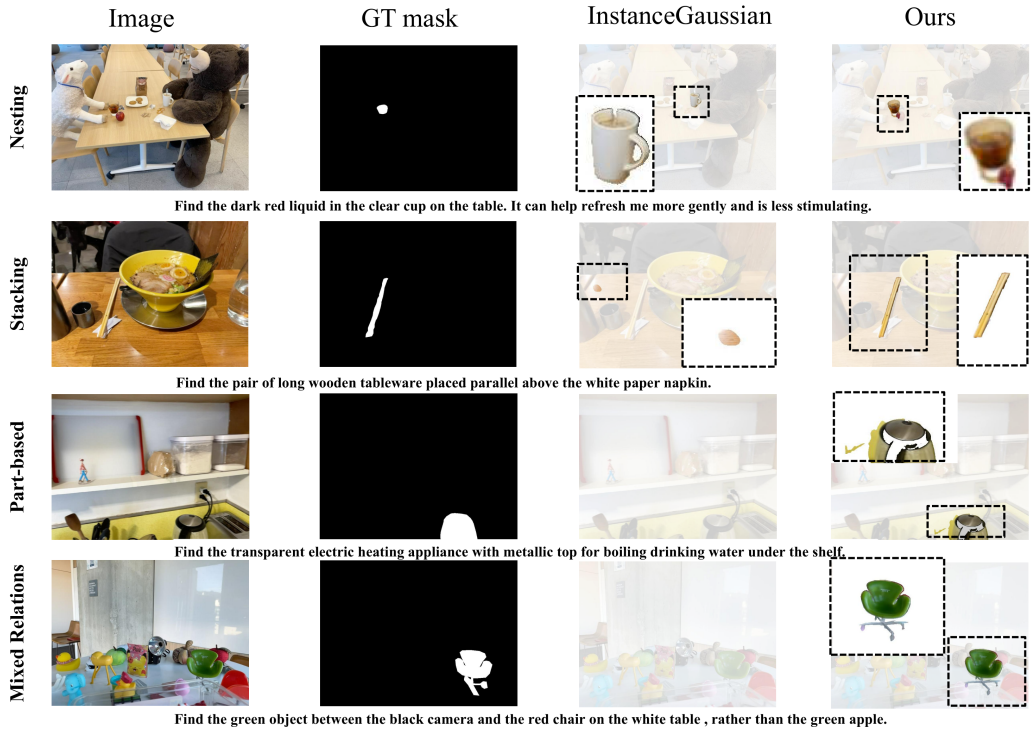


Figure A.1: Qualitative results for the four Nest-LERF relation types. The Nesting row locates dark red liquid inside a transparent cup on the table. The Stacking row locates a pair of parallel wooden chopsticks above a white tissue. The Part-based row identifies an electric kettle from the whole–part relation between its metal top and body, together with its transparent material and position below a shelf. The Mixed Relations row locates a green object from its position on a white table between a black camera and a red chair. White denotes target regions and black denotes background in the ground-truth mask. Prediction columns highlight targets by fading the background; dashed boxes indicate local regions and magnified details.

Target extent in part-based relations. A query may identify a whole through one of its parts, or directly refer to a part or attachment. The kettle example is the former: its metal top provides a component-level cue, while both the query and ground-truth annotation refer to the entire kettle. The segmentation target is therefore the whole object.

NestSplat localizes the queried regions in the liquid-inside-cup and chopsticks-above-tissue examples, whereas InstanceGaussian predicts other objects. Insets for the kettle and green-object examples show prediction boundaries and regions of incomplete coverage.

D.2 Figure A.2: Zero, One, and Grouped Multiple Targets

Figure A.2 presents representative Nest-LERF cases with zero, one, and grouped multiple target instances. They illustrate empty output for a zero-target query, localization of a specified single instance, and coverage of all same-description instances represented by one noun-anchored support in a grouped-multiple-target query. Rows correspond to zero, one, and grouped multiple targets; columns show the original image, CausalSplat, and our method. The complete English query appears below each row, and predictions are overlaid on a faded scene image.

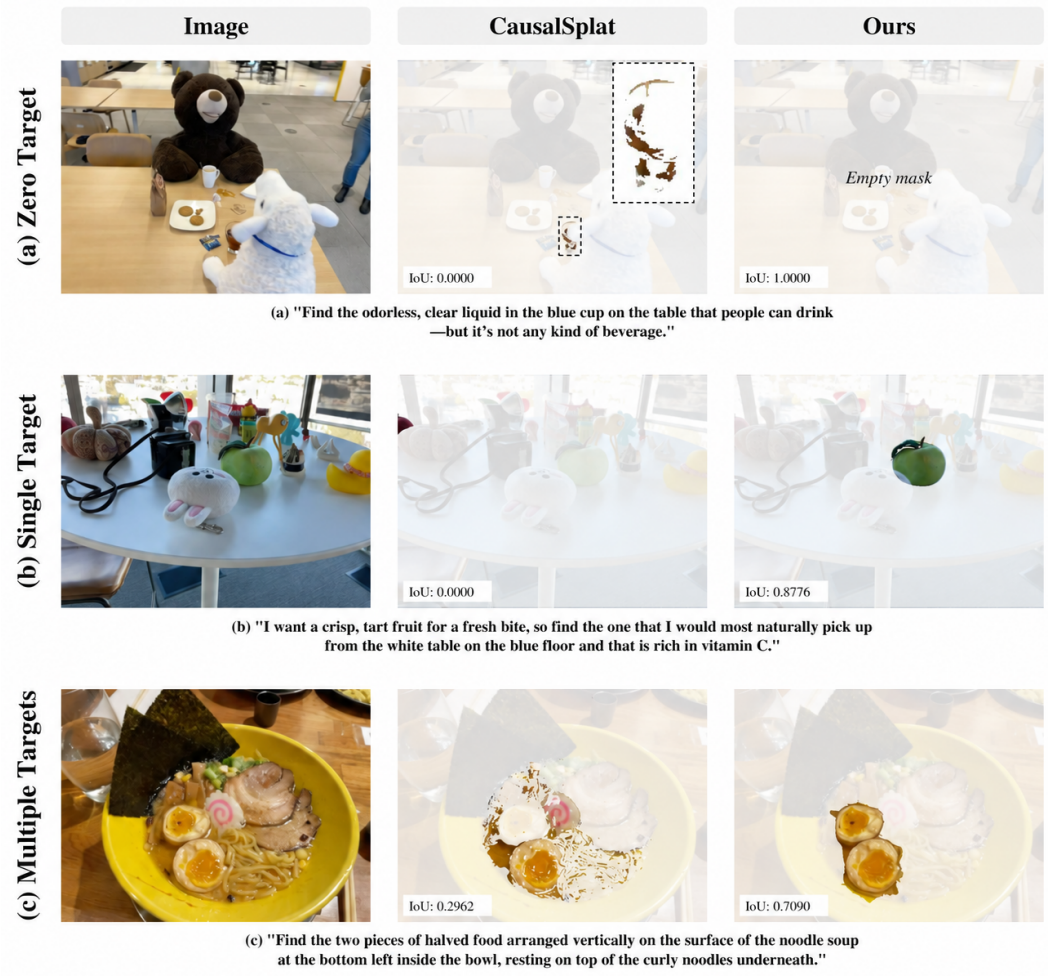


Figure A.2: Nest-LERF results across target cardinalities. Rows show zero, single, and grouped-multiple targets; columns show the input, CausalSplat, and our method. Scores are per query: zero-target cases follow the empty-mask rule, while single and grouped-multiple cases use pixel IoU. The grouped-multiple prediction is produced by one noun-anchored support.

Zero target. In (a), our score is 1.0000 and CausalSplat's is 0.0000. This sample corresponds to `frame_00025` in teatime. The per-query record contains no target node and reports `no_target_empty`, indicating that no node was returned and an empty mask was produced. Figure A.5(a) uses the same instruction in another independently queried test view; its routing decision differs, so the prediction and score differ as well. In Table A.4, NestSplat correctly returns an empty mask for 7 of 26 zero-target samples, for a mean score of 0.2692.

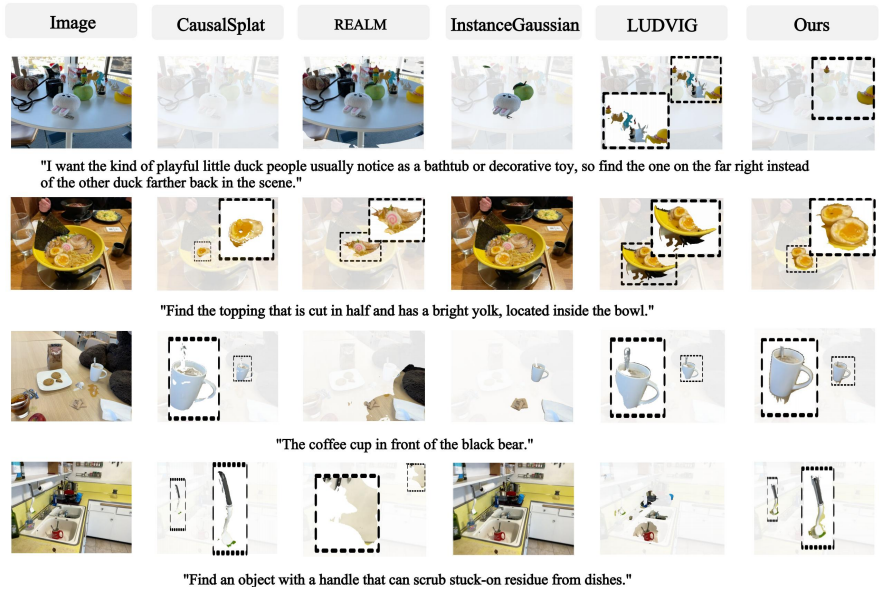
Single target. In (b), our IoU is 0.8776, compared with 0.0000 for CausalSplat. This example shows localization of one instance from a complete natural-language description, contrasting with empty output for zero targets and joint coverage for grouped multiple targets.

Grouped multiple targets. In (c), our IoU is 0.7090, compared with 0.2962 for CausalSplat. Grouped-multiple-target evaluation merges all referred same-description instances into one ground-truth mask. Here, the Gaussian support of one inventory node covers both targets, although boundary errors remain. In Table A.4, the mean grouped-multiple-target scores of NestSplat and CausalSplat are 0.3732 and 0.3617, respectively.

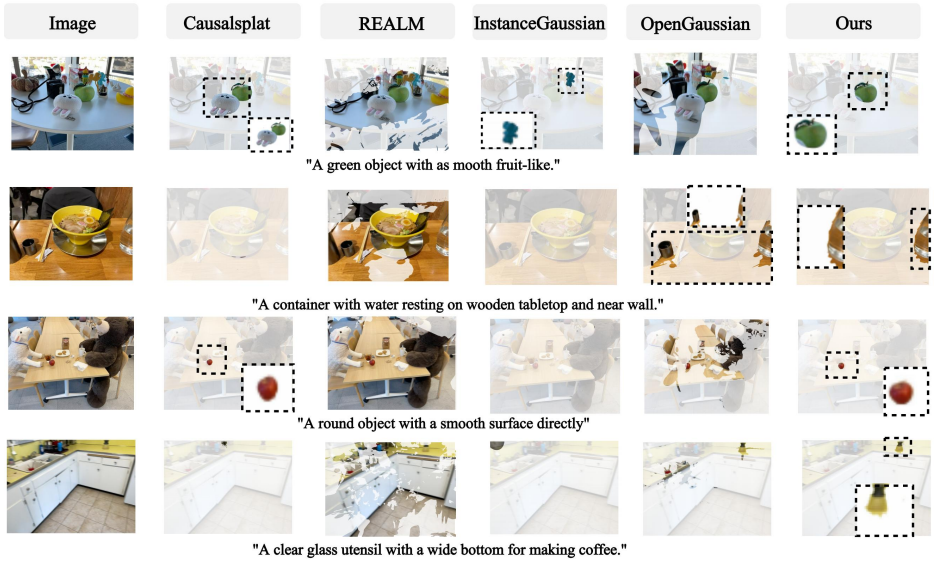
D.3 Additional Qualitative Results on Referring and Reasoning Benchmarks

To conserve main-text space, we collect qualitative comparisons on Causal-LERF, Ref-LERF, and Causal-ScanNet here; the corresponding quantitative results appear in Tables 5, 4, and 6. Figure A.3 compares 2D reasoning and referring queries, while Figure A.4 examines whether functional descriptions are grounded in the correct category and spatial region of an indoor 3D scene.

In Figure A.3(a), the Causal-LERF queries cover same-category instance disambiguation, a target inside a container, a target in front of a landmark, and a functional description. Our predictions concentrate on the corresponding instances. In Figure A.3(b), our method localizes a green apple, a container, a red circular object, and coffee equipment, while several baselines expand into the background, return empty predictions, or select the wrong instance.



(a) Qualitative comparison of query segmentation on Causal-LERF.



(b) Qualitative comparison of query segmentation on Ref-LERF.

Figure A.3: Additional qualitative results on 2D reasoning and referring benchmarks. (a) Causal-LERF queries specify targets through functions, attributes, or spatial relations. (b) Ref-LERF queries cover category, appearance, container, and spatial descriptions. Columns show the input image, baseline predictions, and our results; predicted regions are overlaid on faded backgrounds.

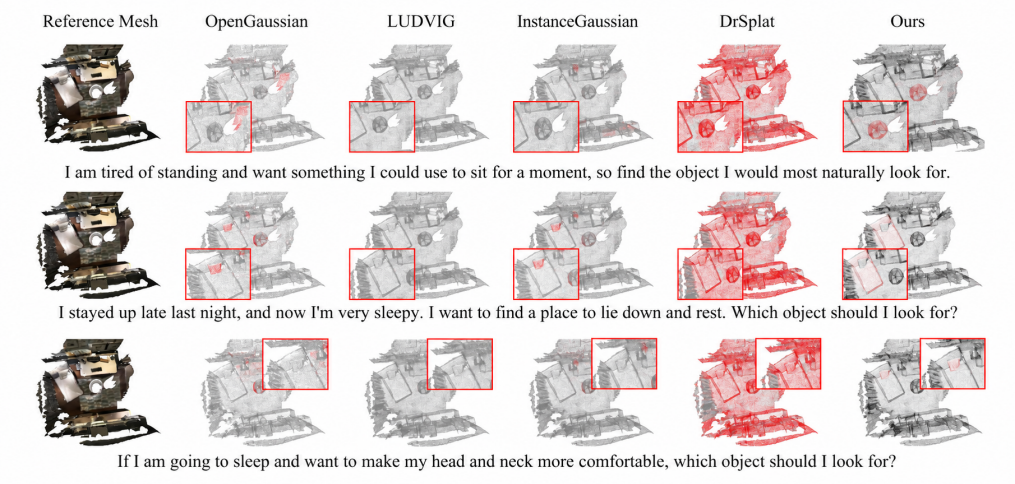


Figure A.4: Qualitative comparison of 3D reasoning segmentation on Causal-ScanNet. Columns show the reference mesh, OpenGaussian, LUDVIG, InstanceGaussian, DrSplat, and our method. The three queries functionally describe a chair, bed, and pillow; colored regions denote predictions.

The functional statements in Figure A.4 do not explicitly name the category. A model must infer the target from its use and then localize it in 3D. Our method confines predictions to the corresponding targets in all three examples, whereas methods such as DrSplat show extensive spurious responses. These examples are consistent with the aggregate trends in Tables 4–6, but are illustrative rather than substitutes for full-test-set statistics.

E Failure Analysis

We analyze three failure modes of NestSplat on Nest-LERF: nonempty predictions for zero-target queries, confusion between similar instances, and confusion between small parts and whole objects. Figure A.5 shows NestSplat predictions. The three queries come from the per-query records used for the main evaluation and correspond to `frame_00002` in teatime, `frame_00162` in waldo_kitchen, and `frame_00083` in figurines.

E.1 Failure to Reject Zero-Target Queries

Only 7 of 26 zero-target queries produce a correct empty mask, showing that the current pipeline may return a nonempty result when the described target is absent. The query in Figure A.5(a) asks for drinkable transparent liquid inside a blue cup. No such combination exists, yet the model selects a white coffee mug and receives IoU 0. This example shows that semantic similarity to drinking is insufficient to satisfy joint constraints on color, container, and contents; selecting a related object cannot replace an explicit existence decision.

E.2 Confusion between Similar Instances

Figure A.5(b) asks for the cup containing two long-handled utensils in the left sink, but the model selects a blue cup in the right sink, yielding IoU 0. In the same frame, both this query and the blue-cup query are routed to `node_1`, revealing a failure to distinguish similar cup instances. Such errors may arise from synonym merging, insufficient inventory granularity, or routing bias.

E.3 Confusion between Small Parts and Whole Objects

The target in Figure A.5(c) is the pink hair tie on the yellow duck at the far right of the white table, but the model selects both the tie and the duck, yielding IoU 0.0868. It fails to separate the small part from the object to which it is attached, expanding the prediction to the whole object.

E.4 Scope of the Additional Geometric Check

Enabling the bounding-box conflict check for LERF tabletop scenes decreases performance on figurines (Table A.8; see Appendix C). That result is an ablation under a modified configuration, whereas Figure A.5 shows failures under the default Nest-LERF configuration.

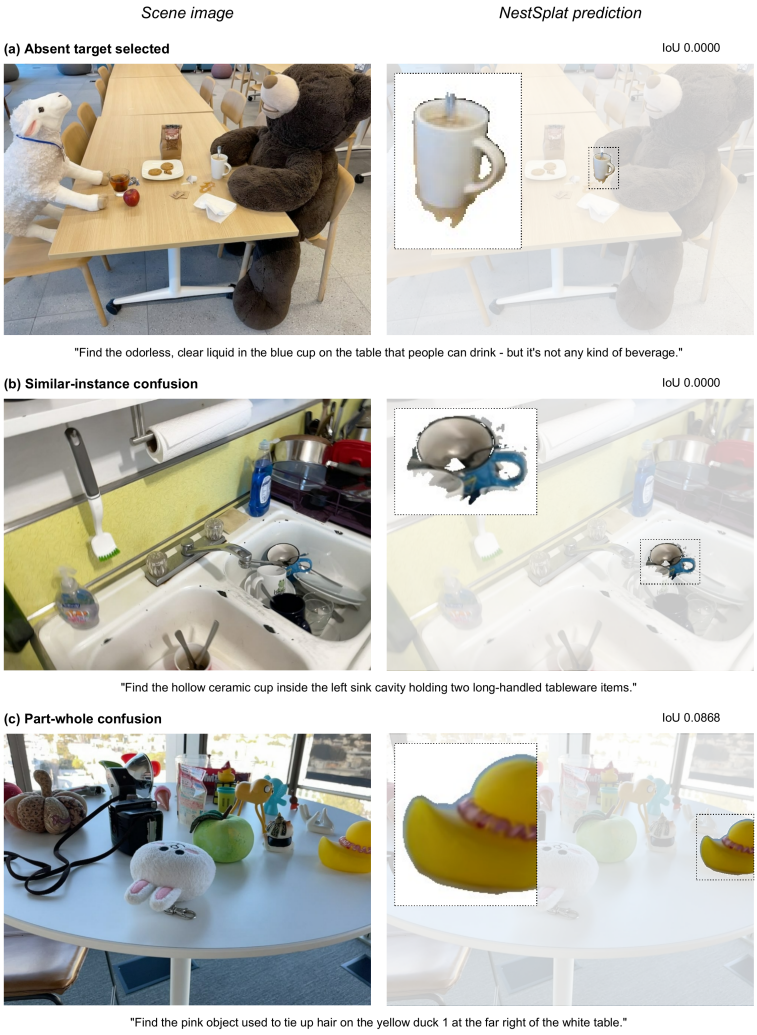


Figure A.5: Failure cases of NestSplat. The left column shows the source scenes and the right column our predictions. Faded backgrounds provide spatial context, and dashed boxes mark predictions and local enlargements. Each row preserves the complete query. (a) A nonexistent combination of transparent liquid in a blue cup instead yields a white coffee mug. (b) A query for the cup in the left sink yields the blue cup in the right sink. (c) A query for the pink hair tie on a duck selects the duck as well. The IoUs are 0, 0, and 0.0868, respectively, all from per-query evaluation records. Prediction panels retain the original colored results; background fading and magnification are for visualization only and do not affect evaluation.

F Efficiency Breakdown

Tables 7–8 report task coverage and efficiency, respectively. This section specifies the measurement scope and timing protocol of one efficiency run on figurines; the environment is described in Section B.1. Scene preparation takes 178.7 min, mean serial wall-clock query time is 6.3 s, and sampled peak memory on the local GPU is 17.8 GiB. Timing includes remote API round trips, while memory excludes remote-model inference resources.

Table A.11: Upper bound on logical VLM calls for the full Nest-LERF benchmark

Item	Calls
Lifting-stage VLM batches ($\leq$48 frames each)	$\sum\lceil N_{\text{train}}/48\rceil = 18$
Text-only inventory merging during lifting	≤ 4
Visual query interpretation	≤ 247
Query-time inventory selection	≤ 247 (omitted under direct selection)
Total	≤ 516

Table A.12: Timing and measurement scope of the efficiency metrics in Table 8

Table 8 Item	Measurement Scope
Scene preparation: 178.7 min	Stage A takes 10,711 s for noun proposal, SAM 3 multi-view masks, contribution lifting, and pre-rendering. Stage B takes 12 s for graph construction and one auxiliary empty-query routing call. Total: 10,723 s (178.717 min), excluding the existing 3DGS reconstruction
Mean serial query time: 6.3 s	The 63 queries in this figurines timing run execute serially. Stage C wall-clock time is 395.1 s, or 6.271 s per query, including amortized initial scene loading, query routing, mask retrieval, and network round trips
Sampled local peak memory: 17.8 GiB	Local GPU `nvidia-smi memory.used` is sampled every 2 s across Stages A–C, including SAM 3. The maximum is 18,233 MiB (17.806 GiB), excluding remote Qwen inference memory

F.1 Upper Bound on VLM Calls

The full Nest-LERF benchmark contains four scenes and 247 queries:

For S scenes and Q queries, the upper bound on logical VLM calls is $\sum_{j=1}^{S}\lceil N_{\text{train},j}/48\rceil + S + 2Q$. With a fixed number of scenes, the complexity is $O(\sum_j\lceil N_{\text{train},j}/48\rceil + Q)$. This bound counts calls required by a normally completed run and excludes retries after API failures and auxiliary calls outside evaluation queries. Geometric graph construction itself invokes no language model. The efficiency run in Section F.3 includes one additional empty-query routing call, which is excluded from the benchmark-query bound in the table but included in measured wall-clock time. The call-count bound does not explicitly depend on graph-node count or the number of Gaussians, although per-call input length, token cost, and local computation may still grow with scene scale.

For each LERF scene, we take the union of test frames from Nest-LERF, Ref-LERF, and Causal-LERF and exclude them from RGB views used for scene preparation. Object-noun generation, multi-view mask generation, and subsequent Gaussian lifting use only the remaining views. The three benchmarks then share the resulting lifted representation and are evaluated on their respective test views. Section B.2 specifies the ScanNet preparation protocol.

F.2 Correspondence to Table 8

Query timing includes approximately 16 s of scene loading and warm-up before the first query. Under the cache described in Section B.5, the 63 queries average 6.3 s each, including amortized loading and warm-up.

Wall-clock timing covers request waiting and network round trips; delays from failed requests and retries would also be included, although no explicit API retry was observed in this run. Mean latency is computed from the total time of all 63 queries in one run. Memory monitoring covers SAM 3 and all subsequent stages, taking the largest sampled device-level usage as the peak. The peak occurred during contribution lifting in Stage A.

All method results in Table 8 are empirical measurements, although timing scope and compute configurations differ. NestSplat preparation and query times follow the protocol above, including waits for remote APIs, while memory measures local GPU usage. Baseline memory values retain the GB units of their measurement records.

Table A.13: Editing operations on Gaussian support

Operation	Figure A.6	Transformation of $\mathcal{P}^*$
Deletion	(a) Object Deletion	Remove Gaussians in the support from $\mathcal{G}$ and re-render, without an additional inpainting network.
Recoloring	(b) Object Recoloring	Modify spherical-harmonic / color coefficients of the support while preserving geometry and opacity.
Translation	(c) Object Translation	Apply a uniform translation to Gaussian centers in the support while preserving scale and orientation.

F.3 Local Computation

Stage B takes 12 s in total: approximately 9 s for graph construction, pruning, and visualization, approximately 2 s for one auxiliary empty-query routing call, and the remainder for stage overhead and timing granularity. The Stage B time is included in the reported scene-preparation time.

G Downstream Scene Editing

The 3D support $\mathcal{P}^*$ of a selected node is a directly editable subset of Gaussians. Using the same training-free lifting and query routing, we demonstrate deletion, recoloring, and translation in the teatime and figurines scenes of Nest-LERF.

G.1 Edit Definitions

At query time, v^* is selected as in Section 3.3. Edits affect only its support $\mathcal{P}^*$, leaving all other Gaussians unchanged.

Editing instructions use short nouns corresponding to inventory entries (`"hand"` and `"apple"` in the figure). Recoloring arrows indicate source and target colors. Side-by-side before/after panels reveal whether supporting surfaces or nearby same-category instances are affected.

G.2 Editing Results

(a) Deletion. We delete `"hand"` in figurines and `"apple"` in teatime. Before/after comparisons reveal residual target content and any effects on the tabletop or neighboring objects.

(b) Recoloring. We change the green apple in figurines to ochre and the red apple in teatime to purple. Color arrows mark source and target colors, and before/after comparisons show whether changes remain confined to the specified target.

(c) Translation. We move `"hand"` in figurines and `"apple"` in teatime to new positions. Before/after comparisons show target integrity and separation from the background. If the target support includes tabletop Gaussians, a local tabletop region may move as well.

G.3 Relation to the Main Task

These examples show that the Gaussian subset selected by query routing can be deleted, recolored, or translated directly, without additional identity encoding or per-scene semantic training. Editing quality depends on target selection and 3D support: incorrect instance selection affects other objects, incomplete support leaves residuals, and background contamination can move non-target regions. We use short nouns here to demonstrate the editing interface; segmentation from longer hierarchical-relation instructions is evaluated by the main benchmarks.

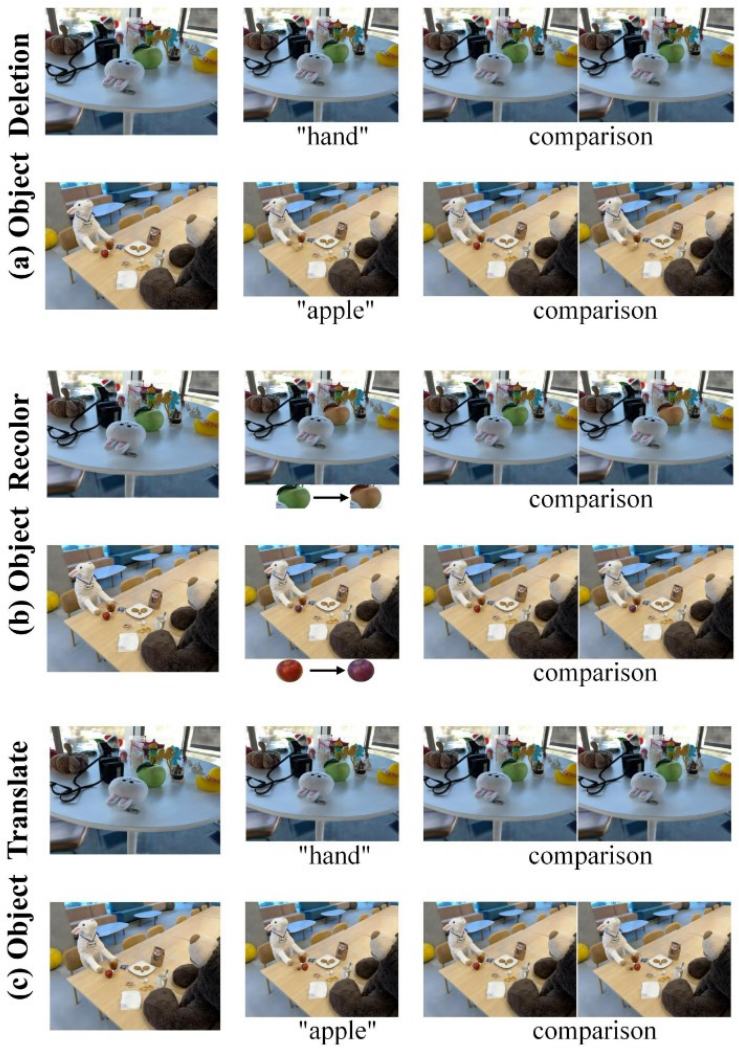


Figure A.6: Language-specified 3D Gaussian editing. The three groups show deletion, recoloring, and translation. In each group, the upper row is figurines and the lower row teatime. The left panels show before/after results and the right panel juxtaposes them for comparison. Target names or color transformations appear below the edited results.

H Real Mobile Capture and Query Segmentation

To complement public benchmarks with an in-the-wild observation, we present language-query segmentation in an indoor scene captured with a mobile phone. The scene contains everyday objects such as wall decorations, curtains, a shopping bag holding plants, and an aquarium. Queries describe targets through appearance, containment, and surrounding landmarks.

H.1 Capture and Reconstruction

The mobile sequence contains 181 frames, of which 128 viewpoints are successfully registered by COLMAP (Schönberger & Frahm, 2016) and used for 3D Gaussian reconstruction. Gaussians are trained for 30k iterations, matching the order of the LERF reconstruction schedule. Because this scene has no independent test-view split, we report only qualitative query-segmentation examples, without novel-view reconstruction metrics or additional segmentation accuracy.

H.2 Queries and Visualization

Figure A.7 shows three queries: a red shopping bag containing plants and other items, a long curtain between a wall decoration and a windowpane, and an aquarium described by cues including its transparent tank and red top panel. The complete English queries are preserved in the figure and correspond to the `red_shopping_bag`, `curtain`, and `aquarium` entries in the evaluation file.

For visualization, target RGB regions are extracted from the captured frame using the predicted mask and overlaid on a faded source image, preserving target location and spatial context. Magnified insets use white backgrounds while retaining mask boundaries and holes.



Figure A.7: Language-query segmentation in a real scene. The top row shows four mobile-phone images of the same indoor scene. From left to right, the bottom row shows queries and predictions for a red shopping bag, a curtain, and an aquarium. Predicted regions retain their original colors while backgrounds are faded. Dashed boxes mark target locations and corresponding magnified regions.

H.3 Scope of the Results

These examples share the public benchmarks' task format of specifying targets through language and visualizing predicted segmentation regions. Because the mobile scene has no independent test-view split, its qualitative results are not combined with public-benchmark test scores. Local coverage gaps are visible, such as the missing region around the small cup above the aquarium; whether that cup belongs to the target extent also depends on how the query distinguishes the whole object from its landmark.