ReLaGS: Relational Language Gaussian Splatting

Supplementary Material

Appendix Overview. This appendix provides additional technical details, implementation notes, ablations, and extended results supporting the main paper. We first present **the architecture of our graph neural network** (Sec. 6), including geometric feature construction, node/edge encoders, the edge-aware transformer, and the contrastive training objective. Sec. 7 describes **implementation details** for Gaussian reconstruction, feature extraction, pruning, language aggregation, and LLM-based annotation. **Additional ablation studies** (Sec. 10) evaluate the impact of pruning, feature aggregation, and GNN components. Sec. 11 details **the evaluation protocol for 3D scene graph prediction**, including object–cluster matching and closed-set readout. We then provide further explanation of our multi-hierarchy querying algorithm (Sec. 12), followed by extended qualitative and quantitative results across all benchmarks (Sec. 13). Finally, Sec. 14 and Sec. 15 summarize datasets and evaluation metrics used throughout our experiments.

6. Graph Neural Network Architecture

To infer open-vocabulary relationships between object pairs, we train a geometry–language fused graph transformer that operates on the 3D scene graph derived from our hierarchical Gaussian representation. Each object (or part) is treated as a node, and each potential relationship forms a directed edge. The goal of the network is to output a 512-dimensional relation embedding for each edge, compatible with the Jina-Embedding-V3 predicate space.

Node and Edge Geometric Feature Construction. Each object node is represented by a 19-dimensional geometric descriptor computed from its oriented bounding box (OBB). Given an object's 3D points, we estimate its center $\mathbf{c}_i$, PCA rotation matrix $\mathbf{R}_i \in \mathbb{R}^{3\times3}$, OBB half-extents $\mathbf{d}_i$, and mass center $\mathbf{m}_i$. These components are concatenated into:

$$\mathbf{g}_i = \left[\mathbf{c}_i \,\|\, \mathbf{m}_i \,\|\, \mathbf{R}_i(:,1) \,\|\, \mathbf{R}_i(:,2) \,\|\, \mathbf{R}_i(:,3) \,\|\, \mathbf{d}_i\right] \in \mathbb{R}^{19}, \tag{12}$$

that captures the object's location, orientation, scale, and height. For each directed edge (i, j), we compute a corresponding relational feature $\mathbf{g}_{ij} \in \mathbb{R}^{19}$ using the OBBs of both nodes. This includes relative displacement $\mathbf{c}_j - \mathbf{c}_i$, Euclidean distance, normalized direction vector, height difference, PCA-axis alignment, and simple OBB overlap/support cues. These geometric features allow the GNN to distinguish relations such as *left*, *behind*, *supported by*, and *inside* purely from spatial structure.

Node and Edge Encoders. Each node fuses its CLIP feature $\mathbf{f}_i$ and geometric descriptor $\mathbf{g}_i$ through a single MLP, and each directed edge (i, j) is encoded by concatenating the two node features with its geometric edge descriptor $\mathbf{g}_{ij}$:

$$\begin{aligned} \mathbf{f}_v^{(i)} &= \mathrm{MLP}_{\text{node}}(\mathbf{f}_i \,\|\, \mathbf{g}_i)\,, \\ \mathbf{f}_{ij} &= \mathrm{MLP}_{\text{edge}}\left(\mathbf{f}_v^{(i)} \,\|\, \mathbf{f}_v^{(j)} \,\|\, \mathbf{g}_{ij}\right), \end{aligned} \tag{13}$$

where $\mathbf{f}_v^{(i)}, \mathbf{f}_{ij} \in \mathbb{R}^{512}$. This provides unified node and edge embeddings that jointly capture semantic and geometric information before graph message passing.

Edge-Aware Graph Transformer. For relational reasoning, we adopt a lightweight edge-aware graph transformer inspired by SGFormer [27]. Unlike the original classification-oriented design, our variant predicts *open-vocabulary relation embeddings* instead of closed-set predicate logits. The transformer operates jointly on node and edge embeddings. At each layer, node updates are computed using *edge-aware attention* in which the edge feature $\mathbf{f}_{ij}$ modulates the attention from node i to node j. This allows the model to incorporate geometric cues directly into the attention weights, improving discrimination of fine-grained spatial relations.

In parallel, edge features are iteratively refined. Each edge embedding receives a gated residual update conditioned on both endpoint nodes and its previous feature, enabling it to accumulate higher-order relational context across layers. Overall, this transformer block follows the same structure as SGFormer—multi-head edge-aware attention, node aggregation, and edge refinement—but adapted to produce continuous 512D embeddings aligned with the Jina predicate space rather than discrete predicate labels.

Training Strategy. We supervise predicted relation embeddings using a simple multi-positive contrastive objective. For an edge (i, j) with predicted embedding $\hat{\mathbf{f}}_{ij}$, ground-truth predicate embeddings $\mathcal{P}$, and a set of sampled negatives $\mathcal{N}$, the loss is:

$$\mathcal{L}_{\text{ctr}} = -\log \frac{\sum_{p\in\mathcal{P}} \exp\left(\hat{\mathbf{f}}_{ij} \cdot \mathbf{f}_p/\tau\right)}{\sum_{p\in\mathcal{P}} \exp\left(\hat{\mathbf{f}}_{ij} \cdot \mathbf{f}_p/\tau\right) + \sum_{n\in\mathcal{N}} \exp\left(\hat{\mathbf{f}}_{ij} \cdot \mathbf{f}_n/\tau\right)}. \tag{14}$$

This loss naturally supports multiple correct predicates per edge and encourages separation from unrelated or antonymic relations. The resulting embedding space captures fine-grained open-vocabulary spatial semantics for 3D scene graph prediction.

Generalization to Gaussian Scenes. Although the network is trained exclusively on 3RScan using point-cloud geometry and RGB-derived CLIP features, it generalizes effectively

to Gaussian scenes because: (i) geometric descriptors are representation-independent, (ii) language embeddings come from CLIP and Jina, which are modality-agnostic, (iii) relational reasoning primarily depends on spatial configuration rather than texture, and (iv) the InfoNCE loss aligns all relation types into a shared embedding space. As a result, the pretrained GNN can be applied directly to our Gaussian scene graphs without any fine-tuning and produces robust relation predictions.

7. Implementation Details

The Gaussian scenes used in all experiments are trained with 2DGS [12] following THGS [7]. For semantic feature extraction, we use SAM ViT-H for 2D segmentation guidance in all experiments, while two types of CLIP are used for different tasks: (1) For open-vocabulary object querying, we follow LangSplat [30] using OpenCLIP ViT-B/16 ($\mathbf{f}_k \in \mathbb{R}^{512}$) as the vision-language encoder. (2) For the task of 3D scene graph prediction and relationship-guided 3D instance segmentation, we follow RelationField to use the vision-language feature from CLIP/ OpenSeg [9] ($\mathbf{f}_k \in \mathbb{R}^{768}$) and use jina-embedding-v3 [35] to encode the language feature of relationships ($\mathbf{f}_{ij} \in \mathbb{R}^{512}$). We introduce two hyperparameters in our hierarchical reconstruction and language lifting pipeline. For Maximum Weight Pruning, we use a contribution threshold $\tau_{contrib} = 5 \times 10^{-4}$, to remove geometrically inconsistent Gaussians without affecting the rendering quality of the scene. For Robust Outlier-Aware Feature Aggregation, we set $\tau_{lang} = 3$ to remove outlier language features. For lifting 3D scene graph from LLM-annotations, we use $K_p = 3$ to pick the top 3 frequent relationships per edge for encoding the open-vocabulary edge embedding. The relationship annotation from 2D images is extracted using GPT-4o [1]. We train our graph neural network on 3DSSG dataset, with all testing sequences in RIO10 subset excluded to prevent data leakage, with a batch size of 4. We begin with a 20-epoch warm-up phase using only the cosine similarity loss, followed by 60 epochs of training with contrastive loss. The network contains only one transformer layer and is trained for 80 epochs with learning rate 1×10^{-4}. We run all experiments of our method using a single NVIDIA RTX 3090 GPU.

8. Hierarchical Query Evaluation on LeRF

A key design goal of our framework is to support hierarchical semantic querying, enabling retrieval at different semantic granularities such as object-level entities (e.g., ramen) and object parts (e.g., noodles). While the LeRF dataset contains queries spanning both levels, the dataset does not explicitly distinguish between object-level and part-level queries. To analyze this capability more precisely, we manually categorize the LERF queries across scenes into object and part queries.

Using this categorization, we evaluate retrieval performance separately for the two groups. The results are summarized in Tab. 8. Our method shows clear improvements over the baseline across both categories, with particularly strong gains on part-level queries. Specifically, our approach improves part-level mIoU by more than **15%**, highlighting the effectiveness of our hierarchical scene representation in capturing fine-grained semantic structure.

Table 8. **Hierarchical query evaluation on LERF.** We manually categorize LERF queries into object-level and part-level queries and report mIoU for each category. Our method improves performance across both groups, with particularly large gains on part-level queries, demonstrating the ability of the proposed representation to support fine-grained hierarchical language queries.

Method	Object mIoU	Part mIoU
THGS	63.2	45
Ours	65.3	**60.3**

9. Additional Analysis on ScanNet

Training Gaussian scenes on ScanNet using a fixed number of primitives leads to suboptimal reconstruction quality, as previously observed in Dr. Splat [14] (see App. C, Fig. S2). Applying Maximum Weight Pruning (MWP) with $\tau_{\text{contrib}} = 0$ removes a substantial fraction of Gaussian primitives, which can further degrade reconstruction quality. To address this, we enable densification during reconstruction (denoted by) and, at evaluation, assign each Gaussian the ground-truth label of its nearest point. Our protocol is conceptually similar to that of Dr. Splat, which uses the Mahalanobis distance; we instead use the standard L2 distance, which simplifies computation while preserving the evaluation behavior. Using this protocol for both THGS and our method, we observe consistent improvements of our approach over THGS across all class subsets (Tab. 9).

Table 9. **ScanNet Results with Revised Evaluation Protocol.** Quantitative results on ScanNet using densified Gaussian reconstruction and nearest-point assignment (L2). THGS* refers to the THGS method re-run by us on all scenes under this protocol. Our method consistently outperforms THGS* across all class subsets.

Method	19 Classes		15 Classes		10 Classes	
	mIoU	mAcc	mIoU	mAcc	mIoU	mAcc
THGS	39.33	54.23	43.61	61.55	52.52	70.50
Ours	41.26	57.29	45.24	62.75	54.04	71.48

10. More Ablation Studies

MWP and ROFA on Scene Graph Prediction. We provide the ablation study of the proposed Max Weight Pruning (MWP) and Robust Outlier-Aware Feature Aggregation (ROFA) methods on the task of 3D scene

graph prediction on 3DSSG [37] dataset in Tab. 10. The graph neural network for 3D scene graph edge prediction relies on the object geometry and semantic feature as input, therefore, accurate the geometry segmentation and language feature lifting significantly improve the object and edge classification results. Our pruning method brings a large performance improvement on 3DSSG dataset, where the original quality of the reconstructed Gaussian scenes is relatively low, due to the low-resolution and motion-blur of the dataset's RGB video sequences.

Table 10. **Ablation** on 3DSSG dataset for 3D scene graph prediction. M. is Maximum Weight Pruning and R. is Robust Outlier-Aware Feature Aggregation .

Variants	Object		Predicate	
	R@5	R@10	R@3	R@5
Base	0.57	0.69	0.75	0.85
Base+$M.$	0.65	0.75	0.77	0.87
Base+$M.$+$R.$	0.68	0.79	0.79	0.87

Graph Neural Network on Scene Graph Prediction. We conduct an ablation study on different design variants of our graph neural network and evaluated them on 3DSSG dataset in Tab 11. As the results show, adding more transformer layers does not improve performance and instead makes the network significantly heavier. This indicates that local information in the scene graph is more important than long-range context for reasoning about relational edges. Furthermore, we also test the different design choices of providing the language feature of nodes to the edge encoder (as in Eq.13) than initializing edge feature from zeros. As the result shows, initializing edge feature from zero performance less good than giving the edge encoder semantic hints from its connected nodes.

Table 11. **Ablation** on 3DSSG dataset for 3D scene graph prediction. L is the number of transformer layer, and $0 \mapsto e$ means using zero vector instead of using node feature pair to initialize edge feature $n \mapsto e$.

Variants	Object		Predicate	
	R@5	R@10	R@3	R@5
L=1, $n \mapsto e$	0.68	0.79	0.79	0.87
L=2, $n \mapsto e$	0.68	0.79	0.77	0.84
L=1, $0 \mapsto e$	0.68	0.79	0.75	0.88

Pruning on Rendering Quality. We conduct an ablation study to examine how maximum weight pruning affects both the rendering quality of the 3D scene and the downstream open-vocabulary segmentation performance. As shown in Tab. 12, applying a light pruning threshold ($\tau_{contrib} = 5 \times 10^{-4}$) significantly improves mIoU from 57.84 to 64.36, while leaving all rendering metrics unchanged. This behavior reflects the role of pruning in removing Gaussians that contribute negligibly to the rendering quality but introduce geometric noise. Eliminating these low-weight Gaussians leads to a cleaner and more accurate scene geometry, which directly benefits the multi-hierarchical clustering step used before lifting language features into the Gaussian clusters.

As the pruning threshold increases, the geometric simplification becomes more aggressive. Moderate thresholds (5×10^{-3} and 5×10^{-2}) begin to slightly perturb the geometry, leading to a gradual decrease in mIoU while still having almost no effect on rendering quality. However, at high thresholds (e.g. 5×10^{-1}), pruning removes a substantial portion of the scene structure, which severely degrades both geometry and photometric fidelity. This collapse propagates to the semantic stage, resulting in a steep drop in segmentation performance. These results demonstrate that lightweight pruning is essential to enhance geometric consistency and, in turn, improve the quality of multi-hierarchical grouping and language feature lifting, while overly aggressive pruning compromises the underlying scene representation.

Table 12. **Ablation Study** on weight pruning.

$\tau_{contrib}$	mIoU ↑	SSIM ↑	PSNR ↑	LPIPS ↓
-	57.84	0.8478	23.576	0.2492
5×10^{-4}	64.36	0.8478	23.576	0.2492
5×10^{-3}	59.27	0.8478	23.576	0.2492
5×10^{-2}	57.53	0.8462	23.526	0.2516
5×10^{-1}	39.18	0.4967	10.905	0.5185

11. 3D Scene Graph Prediction on 3DSSG

Ground Truth and Prediction Alignment. When evaluating on 3DSSG, we face two main challenges. First, our scene graph is constructed on a Gaussian field rather than a fixed, pre-segmented point cloud. In RelationField [19], semantic features can be directly queried at any 3D location within a NeRF, whereas in our Gaussian-based representation, we establish a nearest-neighbor mapping between ground-truth 3D points and their corresponding Gaussians. Second, the ground-truth segmentation in 3DSSG exhibits uneven semantic granularity across scenes. For example, carpets are often merged with floors, while doors and door frames may be labeled separately. To ensure fair comparison, we search across all hierarchical levels of our Gaussian clusters and select, for each ground-truth object, the cluster whose 3D oriented bounding box achieves the highest IoU overlap. After establishing the object–cluster correspondences, we apply our trained graph neural network to predict open-vocabulary relationship features between these clusters.

Closed-set Label Readout. Although our node features $\mathbf{f}_v^{(i)}$

and relation features $\hat{\mathbf{f}}_{ij}$ live in open-vocabulary embedding spaces (CLIP for objects, Jina for relations), inference is performed over a fixed closed-set of dataset labels. For objects, each class name c_k from the closed vocabulary $\mathcal{C}$ is encoded once with the CLIP text encoder to get $\mathbf{t}_k$. Classification is obtained by a cosine-similarity projection, $\text{score}(i,k) = \langle \mathbf{f}_v^{(i)}, \mathbf{t}_k \rangle$, followed by a softmax over k. Similarly, each predicate name r_m from the closed predicate set $\mathcal{R}$ is embedded with the Jina text encoder into $\mathbf{q}_m$. Relation prediction for edge $\mathcal{E}_{(i,j)}$ is obtained by $\text{score}(ij,m) = \langle \hat{\mathbf{f}}_{ij}, \mathbf{q}_m \rangle$, again softmax-normalized across m. In both cases, we take the closed-set ground-truth label whose embedding is most similar to the node/edge feature (in the corresponding CLIP/JINA space) and assign it as the top-ranked label. This provides a clean way to map open-vocabulary features back to dataset-specific closed labels.

Evaluation under Dense Graph Construction. Different from prior works on 3DSSG [18, 19], which typically evaluate triplet recall using fixed thresholds such as Recall@50 or Recall@100, our method constructs a denser spatial scene graph. Specifically, we connect object pairs within a fixed distance threshold (5 m), motivated by the observation that many spatial relationships can potentially exist between nearby objects. However, the human-annotated relations in 3DSSG are relatively sparse, meaning that only a small subset of valid spatial relations is labeled. Under a dense graph construction, the number of candidate triplets grows rapidly with the number of objects and predicate classes, which makes fixed Recall@K metrics less informative. Therefore, in addition to standard Recall@K, we also report **Triplet Recall@1% and 5%** of the ranked predictions. This percentage-based metric normalizes for the size of the candidate relation space and provides a more stable measure of relational reasoning performance under dense graph settings. Using this evaluation, our method achieves a Triplet Recall of **0.86 at 1%** and **0.94 at 5%**, demonstrating strong relational reasoning performance even under a dense graph formulation.

12. Further Detail about Multi-hierarchy Querying

Alg. 1 provides the procedural formulation of the multi-hierarchy querying mechanism described in the main paper. The algorithm implements our key idea that the correct semantic granularity of a text query—whether it refers to a whole object or to one of its finer parts—can be inferred by comparing similarity scores across hierarchy levels. Starting from root-level clusters, the procedure iteratively descends the tree only when child clusters exhibit higher CLIP similarity to the query, thereby adapting the search depth to the level implied by language. To handle cases where several sibling clusters are equally relevant, the algorithm further analyzes similarity drops along the ranked candidates and selects all clusters above the largest drop. The final segmentation mask is obtained by aggregating all Gaussians belonging to the retained clusters, matching the unified object–part reasoning discussed in the main paper.

Algorithm 1: Querying in Hierarchical Scene

Input : Text query q; Cluster features $\{\mathbf{f}_k^{(l)}\}_{l=1}^{L}$; Multi-level labels $\{\mathcal{S}^{(l)}\}$; K of Top-k.
Output : Binary mask $\mathbf{M} \in \{0,1\}^N$ for Gaussians.

1. Encode text query: $\mathbf{t} \leftarrow \text{VLMEncode}(q)$;
2. Get similarity at all levels: $\rho^{(l)} = \cos(\mathbf{t}, \mathbf{f}_k^{(l)})$;
3. Select top-K root-level candidates (l=L) at $\rho^{(L)}$;
4. For each root candidate $S_r^{(L)}$:
  (a) Retrieve its child clusters $\{S_c^{(L-1)}\}$;
  (b) If $\max_c \rho_c^{(L-1)} > \rho_r^{(L)}$, descend one level and repeat (4).
  (c) Otherwise, keep $S_r^{(L)}$ as the matched cluster.
5. Filter out clusters smaller than 1% of parent size;
6. Detect largest score drop $\Delta\rho_{\max}$ and retain clusters above it;
7. Aggregate Gaussians belonging to the selected clusters: $\mathbf{M}[G_i] = 1$ if $G_i \in S_{\text{selected}}^{(l)}$.

Return: hierarchical segmentation mask $\mathbf{M}$.

13. More Results

Open Vocabulary Object Querying on 3D-OVS. We provide more quantitative result for open-vocabulary object segmentation on five scenes of the 3D-OVS dataset [25] in Tab. 13. Following Occam'sLGS [5], we evaluate our method on the corrected annotation of 3D-OVS dataset on the "room" sequence. 3D-OVS dataset mainly contains image sequences of a few objects from close-up viewpoints without much occlusion, in which sharp and clear segmentation of object boundaries plays a critical role for IoU. Although our method based on heuristic clustering performs suboptimal in this set-up and sometimes meets difficulty of providing sharp boundaries, we still achieve a huge improvement compared to THGS and a comparable performance with state-of-the-art methods.

We further provide more qualitative results on LERF, 3D-OVS, ScanNet and ScanNet++ dataset in Fig. 5, on the task of querying single object, object parts and also relation guided object querying. Visualization of the multi-hierarchy scene segmentation and PCA colored language field is given in Fig 6, as well as some examples of the SoM-LLM annotation with our lifting method in Fig. 7.

Table 13. **Open-vocabulary segmentation on 3D-OVS [25].** Methods with $*$ are evaluated on testing-data that has an annotation error in the 3D-OVS testset corrected. † means we run the open-source code under our environment.

Method	3D-OVS mIoU (%)					Mean
	Bed	Bench	Lawn	Room	Sofa	
LangSplatV2 [23]	93.0	94.9	96.1	92.3	96.6	94.6
LAGA [3]	96.8	92.8	97.0	93.0	96.9	95.3
Occam's [5]	96.8	94.8	97.0	96.5*	88.8	95.0
THGS [7]†	68.9	67.6	73.9	68.4*	54.6	66.7
Ours	95.4	94.4	91.2	94.8*	86.9	92.5

14. Datasets

LERF-OVS. LERF [16] introduces language-embedded radiance fields for 2D open-vocabulary localization. LERF-OVS adds object-level masks for quantitative evaluation via 2D mIoU. It mainly tests fine-grained semantic alignment and part–object disambiguation in cluttered scenes.

3D-OVS. 3D-OVS [25] provides RGB-D reconstructions with 3D ground-truth instance labels for open-vocabulary segmentation. Unlike LERF-OVS, it evaluates volumetric 3D semantic consistency. We follow the corrected annotation for the "room" scene as in Occam's LGS.

ScanNet. ScanNet [6] contains 1513 indoor RGB-D scenes with dense 3D semantic labels and is a standard benchmark for 3D semantic segmentation. We follow prior work and report 3D mIoU on the 19/15/10-class subsets.

ScanNet++. ScanNet++ [50] refines ScanNet with higher geometric fidelity, improved trajectories, and more accurate reconstructions. It provides 3D instance annotations and serves as the benchmark for relationship-guided instance segmentation with annotations provided by RelationField[19].

3DSSG. 3DSSG [37] includes 3D semantic scene graphs with 160 object categories and 27 relationship types annotated on pre-segmented point clouds. It is the standard dataset for 3D scene graph prediction. We use the RIO10 subset and follow Open3DSG and RelationField for evaluating object and predicate recall, and for training our lightweight GNN.

15. Evaluation Metrics

Metrics for Open-Vocabulary Querying. For LERF-OVS and 3D-OVS, we evaluate open-vocabulary localization using the standard 2D mIoU protocol from LERF [16] and OVS [46]. Given a query q, each model renders a semantic heatmap from the viewpoint of the ground-truth mask. We compute the per-pixel similarity between the CLIP embedding of q and the rendered language features, threshold it into a binary prediction P_q, and measure:

$$\text{mIoU}(q) = \frac{|P_q \cap G_q|}{|P_q \cup G_q|}, \tag{15}$$

where G_q is the ground-truth 2D region. This 2D mIoU measures how well the language-aligned 3D features project to 2D and localize open-vocabulary concepts at the pixel level.

For 3D benchmarks such as ScanNet [6] and ScanNet++ [50], we instead evaluate semantic correctness directly in 3D. Each Gaussian (or reconstructed 3D point) is assigned a semantic label, and predictions are compared with ground-truth point-level annotations. The 3D mIoU is computed using the intersection-over-union between predicted and ground-truth point sets:

$$\text{mIoU}_{3D} = \frac{|P \cap G|}{|P \cup G|}, \tag{16}$$

where P and G denote the sets of predicted and ground-truth points for a given semantic category. Unlike 2D mIoU, which evaluates pixel-level consistency in single views, 3D mIoU measures volumetric semantic accuracy across the full reconstructed scene. For ScanNet++, predictions are aligned to the ground-truth mesh to compensate for scale and sampling differences before evaluation.

Metrics for 3D Scene Graph Prediction. We evaluate relational reasoning on the 3DSSG dataset [37] following the standard scene graph metrics used in prior work [18, 19, 26, 48]. A scene graph is defined by nodes (objects), edges (relationships), and their semantic labels. Evaluation is performed using three complementary recall-based metrics: **object recall** and **predicate recall**. All metrics report **Recall@K**, i.e., whether the correct ground-truth label appears in the top-K predictions.

Object Recall@K. For each ground-truth object, we compute the cosine similarity between its textual class name and the predicted object embedding. The model outputs a ranked list of object class candidates, and an object is counted as correctly recognized if the ground-truth class appears among the top-K predictions. This evaluates the quality of open-vocabulary language registration and the discriminability of object-level embeddings.

Predicate Recall@K. For each annotated relation edge (s, p, o), the model predicts a ranked list of predicate labels using cosine similarity between the predicted relation embedding and the textual embeddings of all predicates. Predicate Recall@K measures the fraction of ground-truth predicates appearing in the top-K predictions. Because many spatial predicates (e.g., *next to*, *in front of*, *on top of*) may describe similar geometric configurations, this metric captures the model's ability to recognize fine-grained and open-vocabulary.

Triplet Recall@p%. Triplet recall evaluates whether the full relational triplet (s, p, o)—including the subject, predicate, and object—is correctly predicted. For each candidate object pair in the scene, the model produces scores for all predicate classes, forming a ranked list of candidate triplets. All triplets are globally ranked by their confidence

scores, and Triplet Recall@p% measures the fraction of ground-truth triplets that appear within the top-$p\%$ of this ranked prediction set. Reporting recall at a percentage rather than a fixed K better reflects performance in dense graphs, where the number of candidate relations grows with the number of objects and predicate classes.

16. Discussion on Dynamic Scenes and 4D Extensions

While our method focuses on static 3D scenes, the proposed language registration framework can be naturally extended to dynamic environments. In particular, adapting the approach to 4D scene representations would require incorporating temporal consistency into the clustering and relational reasoning mechanisms used in our pipeline.

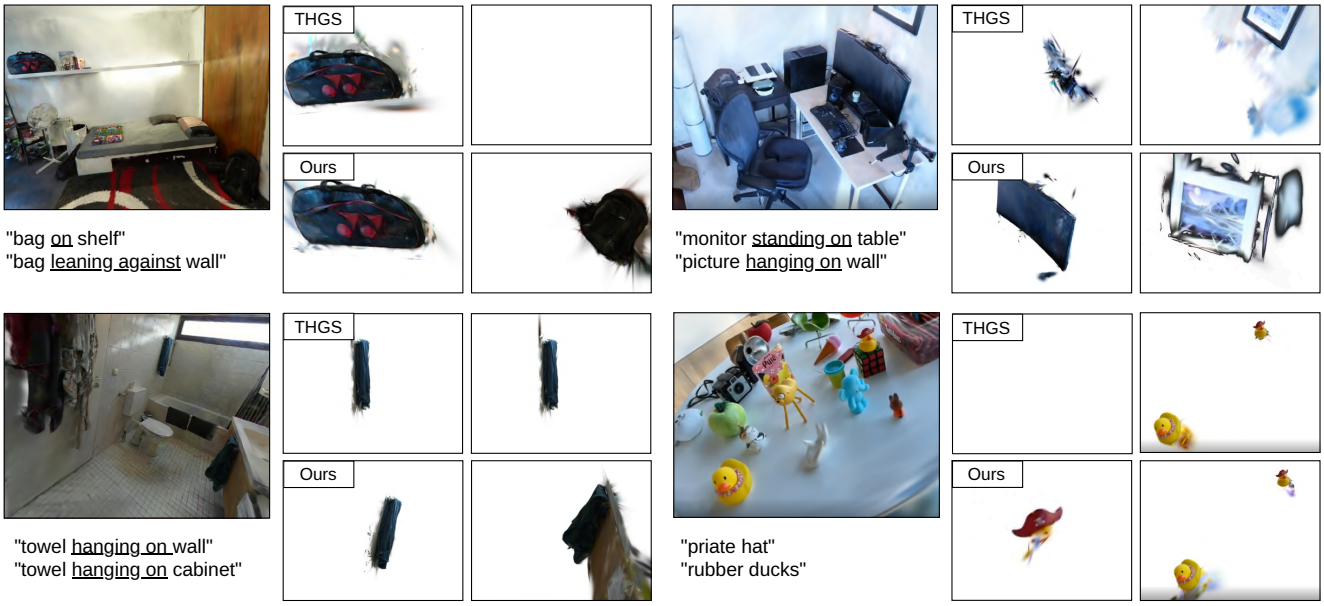
A key component of our method is the construction of a KNN graph over Gaussians, which is subsequently used for hierarchical clustering and relational reasoning. In dynamic scenes, this graph must account not only for spatial proximity but also for temporal consistency across frames. The precise adaptation depends on the type of dynamic Gaussian representation used.

In approaches where scene dynamics are modeled through deformation networks applied to a fixed set of Gaussians, the number of Gaussians remains constant over time. In this setting, extending our method would primarily require redefining Gaussian proximity across time. Specifically, Gaussians belonging to the same object should maintain spatial consistency throughout the time intervals in which they are visible. Under this assumption, the KNN graph can be constructed using a temporal aggregation of spatial distances, allowing the hierarchical clustering procedure to remain largely unchanged. Importantly, our language registration components, including MWP and ROFA, can be applied without modification.

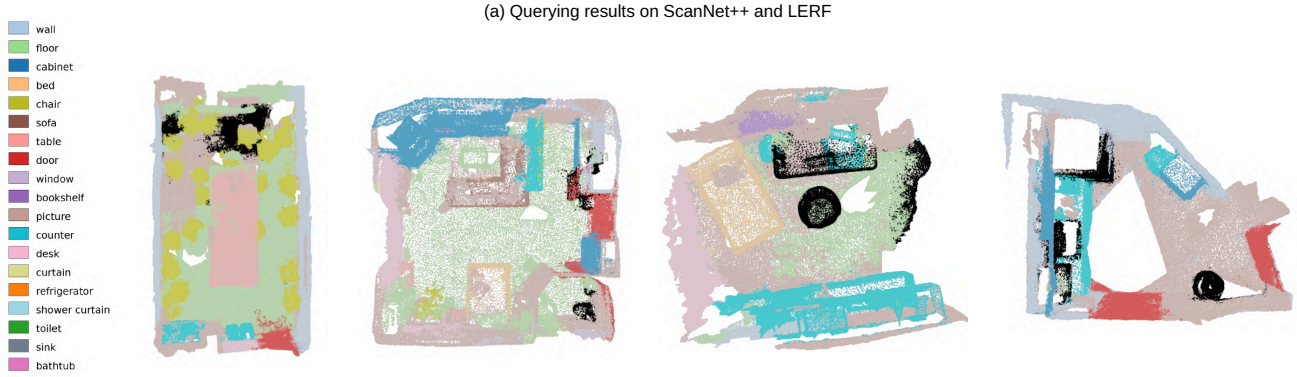
A second class of dynamic representations models scenes using 4D Gaussians, where primitives may appear or disappear over time. In this case, the graph construction must explicitly incorporate the temporal dimension, as spatial proximity alone is insufficient to determine relationships between primitives. Both the KNN graph and the subsequent hierarchical clustering would therefore need to operate in a joint spatio-temporal space, where edge weights reflect proximity across both spatial and temporal dimensions.

Finally, dynamic scenes introduce time-varying relationships between objects, which cannot be fully captured by a static scene graph. Extending our relational reasoning framework would therefore require the use of a dynamic scene graph that evolves over time, enabling the model to represent changing object interactions and relations.

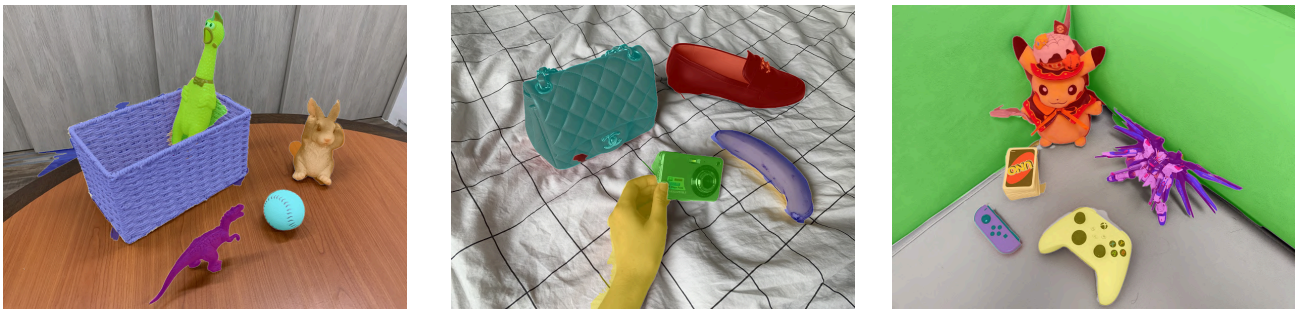
Overall, these considerations suggest that the proposed framework provides a promising foundation for language-grounded reasoning in dynamic 4D scenes, with the primary extensions involving temporally-aware graph construction and clustering while leaving the core language registration mechanisms largely unchanged.



(a) Querying results on ScanNet++ and LERF



(b) 3D Semantic segmentation results on ScanNet



(c) 2D open-vocabulary object segmentation on 3D-OVS

Figure 5. **Qualitative results** on LERF, ScanNet++, ScanNet and 3D-OVS.

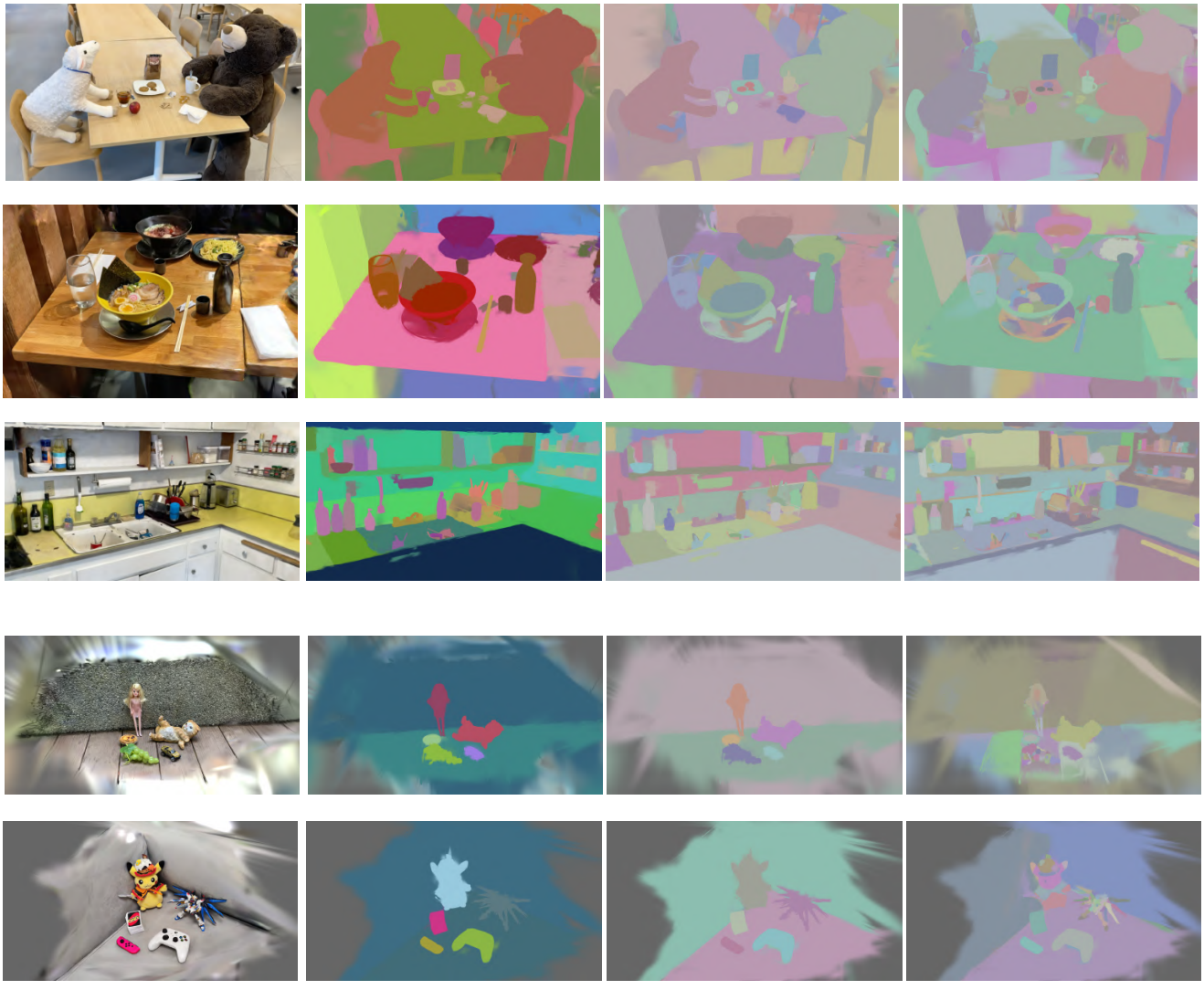
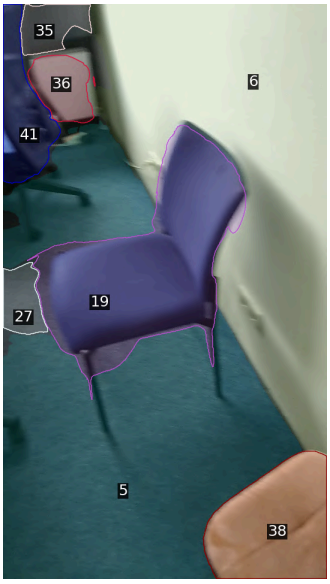
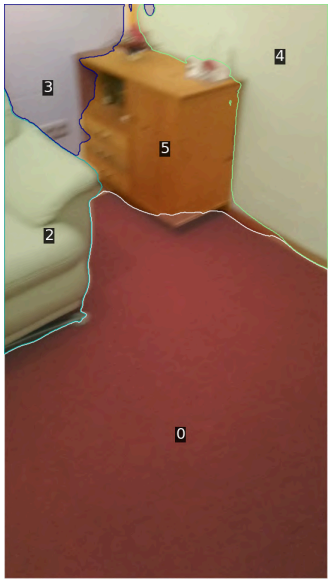


Figure 6. Visualization of multi-hierarchy scene reconstruction on LERF and 3D-OVS datasets.


```
"objects": {
    "1": "floor",
    "16": "laundry basket",
    "17": "wardrobe",
    "25": "wall",
    "29": "bin",
    "31": "clothes",
    "38": "table",
    "42": "clothes"
},
"relationships_affordances": [
    {
        "s_id": 42,
        "subject_class": "clothes",
        "o_id": 16,
        "object_class": "laundry basket",
        "predicates": "inside, contained in"
    },
    {
        "s_id": 29,
        "subject_class": "bin",
        "o_id": 1,
        "object_class": "floor",
        "predicates": "above, standing on"
    },
    {
        "s_id": 17,
        "subject_class": "wardrobe",
        "o_id": 1,
        "object_class": "floor",
        "predicates": "above, standing on"
    },
    ......
]
```


```
"objects": {
    "5": "floor",
    "6": "wall",
    "19": "chair",
    "27": "desk",
    "35": "monitor",
    "36": "computer",
    "38": "chair",
    "41": "chair"
},
"relationships_affordances": [
    {
        "s_id": 19,
        "subject_class": "chair",
        "o_id": 5,
        "object_class": "floor",
        "predicates": "above, standing on"
    },
    {
        "s_id": 38,
        "subject_class": "chair",
        "o_id": 5,
        "object_class": "floor",
        "predicates": "above, standing on"
    },
    {
        "s_id": 27,
        "subject_class": "desk",
        "o_id": 5,
        "object_class": "floor",
        "predicates": "above, standing on"
    },
    .....
]
```


```
"objects": {
    "0": "floor",
    "2": "couch",
    "3": "wall",
    "4": "wall",
    "5": "cabinet"
},
"relationships_affordances": [
    {
        "s_id": 2,
        "subject_class": "couch",
        "o_id": 0,
        "object_class": "floor",
        "predicates": "above, standing on"
    },
    {
        "s_id": 5,
        "subject_class": "cabinet",
        "o_id": 0,
        "object_class": "floor",
        "predicates": "above, standing on"
    },
    {
        "s_id": 5,
        "subject_class": "cabinet",
        "o_id": 3,
        "object_class": "wall",
        "predicates": "next to, attached to"
    },
    ".....
]
```



```
"objects": {
    "0": "floor",
    "1": "carpet",
    "3": "towel",
    "6": "towel",
    "8": "bathtub",
    "13": "wall",
    "15": "curtain",
    "19": "bathtub"
},
"relationships_affordances": [
    {
        "s_id": 1,
        "subject_class": "carpet",
        "o_id": 0,
        "object_class": "floor",
        "predicates": "above, lying on"
    },
    {
        "s_id": 0,
        "subject_class": "floor",
        "o_id": 1,
        "object_class": "carpet",
        "predicates": "below, supporting"
    },
    {
        "s_id": 6,
        "subject_class": "towel",
        "o_id": 8,
        "object_class": "bathtub",
        "predicates": "above, lying on"
    },
    ......
]
```


SoM frame LLM annotation SoM frame LLM annotation

Figure 7. Examples of our SoM+LLM scene graph annotation on 2D images from 3DSSG dataset.