CVPR #15

Appendix

A. Implementation Details for Architecture

A.1. Object-level Feature Extraction

An essential aspect of our feature extraction pipeline is object-level patch feature integration, which enriches relationship queries with fine-grained visual information. We explore three types of visual encoders: the CLIP image encoder, DINOv2-pretrained features, and the original SigLIP encoder used in LLaVA-OV. As shown in Table 9, the SigLIP features yield the most significant performance improvement, likely due to their stronger image-text alignment learned during pretraining. The benefit of using SigLIP is that it serves as the original vision encoder for LLaVA-OV, which not only ensures better feature alignment but also avoids introducing an additional vision encoder. For each object in a triplet, we use its downscaled ($384 \times 384 \rightarrow 27 \times 27$) 2D bounding box to query the corresponding image patch features. If a feature patch overlaps with the object's 2D region, it is included in the computation. We then average the selected patch embeddings to form the object-level feature, which is concatenated with the corresponding text embedding.

Table 9. **Ablation study of patch visual model variants.** We evaluate the impact of different visual models on the object patch features on relation prediction accuracy.

Vision Encoder	VG (In-domain 5K)	PSG (Cross-domain 1K)	
	Acc↑	Acc↑	Acc w/ Similarity↑
***CLIP** [35]*	46.2	22.8	57.3
***DINOv2** [32]*	46.2	22.5	57.7
***SigLIP** [59]*	**46.3**	**23.6**	**59.2**

A.2. Depth Exploration

While our method primarily relies on 2D visual and semantic cues, we explore the possibility of incorporating depth information into the relation prompting process, motivated by recent works such as VCoder [14]. Unlike approaches that use explicit 3D positional embeddings or train a specialized 3D encoder [64], requiring large-scale supervision to align with language space, we instead adopt an efficient strategy that reuses the pretrained visual encoder already aligned with the text modality for depth, as Sec. A.1. Specifically, we estimate monocular depth maps from 2D images using Depth Anything v2 [52], and normalize the predicted depth values to the [0,1] range for consistency. For inference, the model can be seamlessly used with ground-truth depth or estimated depth. After obtaining the depth map, we feed it directly into the frozen image encoder. The object-level depth features could be extracted by pooling visual encoder patch embeddings corresponding to each object's region in the depth map, as the image embedding. They are concatenated after the <DEPTH> placeholder for each object, as shown in Fig. 4, and then fed into the backbone.

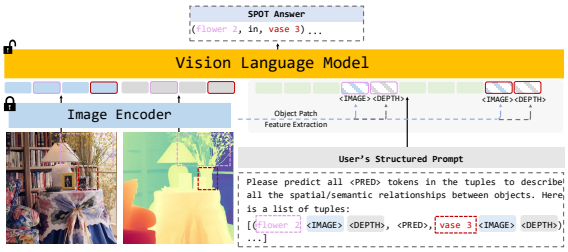

Figure 4. **SPOT method with depth for input**

Similarly to object-centric image embedding, we also explore different intuitive alternatives for depth embedding and compare quantitatively in Tab. 10. The exploration ranges from 3D positional embedding, an additional 3D encoder, and the reuse of the visual encoder, as in our model. For the 3D positional embedding, the depth is lifted into the point cloud to extract the bounding box minimum and maximum coordinates along the z-axis, and we use a fixed sinusoidal positional encoding to embed these 3D locations. For the additional 3D encoder, we use the same method to lift the point from 2D to 3D for the whole object point cloud, and use the pretrained Uni3D [64] encoder for embedding. An additional projection layer is trained to align the input space. As observed in Tab. 10, encoding the depth map with the VLM image encoder performs better. We hypothesize this is because the image encoder is already aligned with the LLM input space, and the depth patches are naturally aligned with the image patches since they are encoded by the same model.

Table 10. **Ablation study of different depth integration variants.**

Depth Info Variant	VG (In-domain 5K)	PSG (Cross-domain 1K)	
	Acc↑	Acc↑	Acc w/ Similarity↑
3D Positional Embedding	46.3	23.2	58.4
***3D Encoder** [64]*	46.3	22.6	59.1
***SigLIP Depth** [59]*	**46.4**	**23.6**	**59.5**

Empirically, given the above analysis, we observe that the inclusion of depth features yields marginal improvement in quantitative metrics. However, from a qualitative perspective, the model produces better spatial relations as shown in Fig. 13. While the improvements brought by depth are not quantitatively reflected in our current evaluation settings, we believe it presents a promising direction for future work on explicit spatial reasoning. Hence, we provide the depth integration option here for further development and investigation.

CVPR #15

B. Implementation Details for Training

B.1. Data Preprocessing

In our experiments, we finetune the model on Visual Genome 150 [20], which serves as the primary dataset to ensure fair comparison with other baseline methods presented in the main paper. Input images are preprocessed to match the expectations of the SigLIP vision encoder. All images are resized to a resolution of 384×384 and normalized using the standard mean and standard deviation values from the model's pretraining configuration. We do not apply any data augmentation techniques such as random flipping during the fine-tuning stage in order to maintain a consistent mapping between visual cues and relational language. Bounding box coordinates are extracted inside every object pair and normalized to a range of [0, 1] relative to the image dimensions before being used for object-centric feature extraction.

B.2. Structured Prompt Details

Fig. 5 illustrates our structured prompt template with example input. Instead of sequentially querying each object pair for their relationship, we adopt a more efficient approach by constructing a structured prompt template, illustrated in Figure 5. Within this template, <IMAGE> (and <DEPTH> added alternatively) are fixed special tokens that serve as anchors for inserting object-level visual (and depth) features, as described in Section A.1. These tokens are not updated during training but are used to locate where the visual embeddings should be injected. We also introduce a special <PRED> token to indicate the position where the model should predict the relationship between the object pair. This design allows for explicit supervision and reduces redundancy in the prediction process. Additionally, each object name is followed by a numeric identifier to distinguish between different instances of the same class.

```
{
"from"  "human",
"value"  "<image>\nYou are an agent specializing in identifying the physical and
spatial relationships in images for 3D mapping.\nYour task is to analyze the
images and output a list of tuples describing the physical relationships between
objects.\nNote that you are describing the **physical relationships** between
the **objects inside** the image. \nYou will also be given a text list of
relation tuples. The list will be in the format: [("object 1", "<PRED>", "object
2"), ("object 3", "<PRED>", "object 2")].\nPlease predict all <PRED> in the
tuples.\nHere is the list of predicate tuples: [(bench 0 <IMAGE>, <PRED>, grass
0 <IMAGE>), (sky 4 <IMAGE>, <PRED>, grass 6 <IMAGE>),..., (wall 8 <IMAGE>,
<PRED>, building 7 <IMAGE>)]. Please describe the spatial relationships between
the objects in all tuples."
}
{
"from"  "gpt",
"value"  "[(bench 0, on, grass 6), (sky 4, over, grass 6), (sky 4, over,
building 7),...,(wall 8, enclosing, building 7)]"
}
```

Figure 5. SPOT structured prompt template.

B.3. Hyperparameter Tuning

During training, the vision encoder is kept frozen, while the language model and adapter layers are trainable. We do not apply AnyRes during either training or inference. The model is optimized using Adam with a learning rate of 1×10^{-5}, a cosine learning rate scheduler, and a warm-up ratio of 0.03. Training is conducted on 8 NVIDIA A40 GPUs with a batch size of 16 for 1 epoch. Due to out-of-memory issues commonly encountered with long prompt sequences, we limit the model's maximum token length to 5000. All other hyperparameters and training configurations follow those of LLaVA-OV-7B.

C. Implementation Details for Inference

In this section, we specifically introduce how our model works during inference to supplement the outline in Section 3.3 in the main paper. There are two key design points to meet our expectations: (1) The pipeline should work robustly for open-world applications with broad generalization abilities. (2) The pipeline should have the ability to filter and rank noisy relations. To satisfy these two requirements, SPOT disentangles relation pruning and proposal from direct relation prediction, leaving the VLM module only the task for spatial reasoning. The relation proposals are provided by out-of-the box detection models, leveraging the most recent advancements in this area. We observed that using this simple but effective approach, our framework could detect fine-grained relations and could be seamlessly applied to different scenarios, providing more comprehensive graphs compared to directly using VLMs to process everything. However, this framework raises three challenges:

- The quantity of detected bounding boxes is generally too large.
- The relation proposal sequences are overly long, making it inefficient for VLM to output long texts.
- There are many spatially implausible object pairs existing.

To mitigate these problems, we apply some strategies to improve the overall performance

Over-detection Suppression. In detection results, multiple bounding boxes may correspond to the same real-world object but have different yet reasonable category labels. For instance, a person may be detected as both "man" and "person," and both exist in the vocabulary. To mitigate this, we apply cross-category Non-Maximum Suppression (NMS) apart from the standard NMS to reduce overlapping boxes across semantically similar labels. For each remaining object, we aggregate all plausible labels into a label group. During evaluation, if the ground-truth label is present in this group, the object is considered correctly classified. For the real application, any label works for usage.

Filter and Rank Relations. To address this problem, we incorporate the classification probabilities from the object detector as object-level confidence scores, selecting the first 100 objects of higher probabilities. For relation-level filtering, we compute the distance between the centers of two objects. We explored three different types of distance for reference.

1 Standard normalized geometry distance: $d = \frac{||c_i - c_j||_2}{\sqrt{H^2+W^2}}$,

2 Standard normlizaed 3D spatial distance: $d = \frac{||c_i^{3D} - c_j^{3D}||_2}{\sqrt{H^2+W^2+Z^2}}$,

3 Distance considering the object size: $d = \sqrt{d_{geo} d_{size}}$, where $d_{geo} = \frac{||c_i - c_j||_2}{\sqrt{H^2+W^2}}$ and $d_{size} = \frac{||c_i - c_j||_2}{||c_i - c_j||_2 + (d_i + d_j)/2}$,

where c_i and c_j are the 2D center points; c_i^{3D} and c_j^{3D} are the 3D center points; H, W and Z denote the height, width and depth of the image; d_i and d_j are diagonal of object bounding boxes. We observe that the third distance works better than the others, and [1], [2]'s results are similar. The score for a candidate pair (i, j) is defined as: $s_{ij} = p_i \times d_{ij} \times p_j$, where p_i and p_j are the classification probabilities, and d_{ij} is the normalized center distance. We discard relation pairs with $s_{ij} > 0.8$, as they are likely too far apart to form meaningful relations.

D. Implementation Details for Benchmark Evaluation

D.1. Overview of Evaluation Setting

We follow the evaluation protocol established in [47], adopting two commonly used settings: Predicate Classification (**PredCLS**) and Scene Graph Detection (**SGDet**). In PredCLS, both ground-truth object categories and bounding boxes are provided. The model receives candidate object pairs and predicts their relationships. In SGDet, the model must operate on detected objects rather than ground-truth annotations. This setting is more reflective of real-world deployment, as it introduces potential detection errors that affect downstream relation prediction. The primary distinction between PredCLS and SGDet lies in the source of object detections—ground-truth versus predicted. A key requirement in these settings is the ability to rank and filter predicted relation triplets, prioritizing the most informative and plausible ones. We evaluate this using Recall@k, where k denotes the number of top-ranked predictions considered. Additionally, to assess the overall relation prediction accuracy, we compare the model's full predicted scene graph against the ground-truth graph. In our main paper, we report both PredCLS and SGDet results for closed-set evaluation on VG150. For cross-domain evaluation and 3D whole-scene evaluation, we focus on PredCLS, where object annotations are available to isolate relation prediction performance. The evaluation setting remains consistent inside every table.

D.2. Baseline Selection and Adaptation

In-Domain Evaluation. We evaluate a range of baseline models, which can be broadly categorized into VLM-based and non-VLM-based approaches. For non-VLM-based models, we compare against methods trained on the same dataset to ensure a fair comparison. These models typically rely on task-specific architectures and do not incorporate large pre-trained vision-language models. For VLM-based baselines, we compare with another recent work PGSG [24], which relies on BLIP as backbone and uses free-form prompt: "Generate the scene graph" in the paper.

Cross-Domain Evaluation. We compare SPOT with other open-vocabulary approaches on new domain data which has never been seen during the training to explore and compare the open-world generalization abilities. For the VLM baselines like LLaVA-OV-7B and GPT-4o, we use the SPOT structured prompt with the addition of an in-context example output (shown in Fig. 6). For ConceptGraphs, we use their free-form prompt template (Fig. 7), with the removal of the original prompt's fixed set of relation options in order to adapt to the wider set of relations present in 3DSSG (see Fig. 6). The visual context is provided by overlaying bounding boxes on the image to indicate object locations, ensuring consistent visual grounding across models. For the other previous approaches like OvSGTR, we directly apply them to the new-source inputs with their built-in end-to-end detection and ranking modules.

```
{
"from"  "human",
"value"  "<image>\nYou are an agent specializing in identifying the physical and spatial relationships in images for 3D mapping. \nYour task is to analyze the images and output a list of tuples describing the physical relationships between objects.\nNote that you are describing the **physical relationships** between the **objects inside** the image.\nYou will also be given a text list of relation tuples. The list will be in the format: [(\"object 1\", \"object 2\"), (\"object 3\", \"object 2\")].\nPlease predict all relations in the tuples.\nHere is the list of predicate tuples: [(bench 0, grass 6), (sky 4, grass 6), ..., (wall 8, building 7)]. Please describe the spatial relationships between the objects in all tuples. \nAn illustrative example of the expected response format might look like this: [("object 1", "on top of", "object 2"), ("object 3", "under", "object 2")].\nOnly output the relation triplet list without additional explanation and symbols."
}
{
"from"  "gpt",
"value"  ""
}
```

Figure 6. Prompt template similar to SPOT for baseline evaluation

```
{
"from"  "human",
"value"  "<image>\nYou are an agent specializing in identifying the physical and spatial relationships in images for 3D mapping. \nYour task is to analyze the images and output a list of tuples describing the physical relationships between objects.\nNote that you are describing the **physical relationships** between the **objects inside** the image.\nYou will also be given a text list of the numeric ids of the objects in the image. The list will be in the format: [\"name1 1\", \"name2 2\", ...].\nAn illustrative example of the expected response format might look like this: [(\"object 1\", \"on top of\", \"object 2\"), (\"object 3\", \"under\", \"object 2\")].\nHere is the list of labels for the annotations of the objects in the image: ['bench 0', 'bench 1', 'tree 2', 'fence 3', 'sky 4', 'pavement 5', 'grass 6', 'building 7', 'wall 8']. Please describe the spatial relationships between the objects in the image. \nOnly output the relation triplet list without additional explanation and symbols."
}
{
"from"  "gpt",
"value"  ""
}
```

Figure 7. Prompt template in free form, similar to ConceptGraphs for baseline evaluation

D.3. Relation Evaluation

In-Domain Evaluation. Since the model is directly trained on the same dataset, the results can be evaluated without any vocabulary gap as in prior works. We use recall to measure how many ground truth relations are correctly predicted, without accounting for inverse or symmetric variants. Specifically, when the ground truth includes only one directional relation (e.g., "A under B" without the inverse "B under A"),

CVPR #15

```
Prediction: [<mouse3> <next to> <keyboard 4>]
Ground Truth: [<mouse3> <beside> <keyboard 4>]

Prompt: Given (mouse next to keyboard), (mouse beside keyboard),
whether these two phrases indicate similar spatial relations between
two objects inside the the tuple. Only return Yes or No without
further explanation.

Answer: Yes
```

Figure 8. LLM as a judge to evaluate the similarity between two relation words

the results are evaluated strictly based on the directional relation that is present in the annotations. If the model predicts the inverse (e.g., "B under A"), it is not considered correct unless it exactly matches the ground truth. While this setting ensures consistency with the prior work and fair comparison, we acknowledge that extending the evaluation protocol to account for equivalent relationships in the opposite direction is an interesting future work to explore.

Cross-Domain Evaluation. A common limitation in existing evaluation protocols for cross-domain datasets is that semantically similar but lexically different relations are treated as incorrect. For instance, synonymous expressions like "beside" and "next to" may convey the same meaning but only one may appear in the ground truth, leading to unfair penalization. To address this, we introduce a more semantically-aware evaluation method by leveraging a large language model (Qwen-7B) as a judge. Given a relation prediction and its corresponding ground-truth triplet, we prompt the LLM with a predefined template (shown in Figure 8) to determine whether the predicted relation is semantically equivalent to the ground truth. If the model answers yes, the prediction is considered correct. This enhanced evaluation metric is reported as "Recall w/ Similarity" or "Accuracy w/ Similarity". This evaluation protocol aims to give model credit for correct relations expressed in an open vocabulary that are semantically equivalent to the ground truth.

The whole cross-domain evaluation pipeline is as follows, including some specific rules for simple cases:

- **Case 1:** prediction relation == ground truth relation → correct
- **Case 2:** prediction contains the same spatial word as the label but with the inclusion of "on", "in", or "from" → correct:
  - In our experiments on the 3DSSG dataset, we use "standing on/on", "built in/in", "hanging on/hanging from".
  - While these rules account for common cases that are clearly correct, it is difficult to iterate all plausible rules comprehensively. Thus, for all other cases, we propose to use LLM as a judge in order to assess the accuracy of the model's open vocabulary predictions against a closed vocabulary label space.
- **Case 3:** otherwise, we use LLM (Qwen-7B) to judge. The prompt is as Fig. 8:
  - We checked the results of LLM's judgment to verify its ability to judge semantically equivalent spatial phrases and found it to be accurate in practice. Given its strong performance and flexibility, we adopt the LLM-based judgement for all cross-domain tasks.

E. 3D Scene Graph Generation pipeline

E.1. Pipeline Overview

To enable scalable and generalizable 3D-SGG in open-world settings, we follow the pipeline introduced in ConceptGraphs [10], leveraging large-scale 2D foundation models for both object discovery and relation reasoning. The overall process consists of two core stages: (1) 3D object construction, which defines the graph nodes, and (2) relation prediction, which defines the graph edges.

Formally, we aim to construct a 3D scene graph $G = (V, E)$, where $V = o_1, o_2, ..., o_N$ is the set of 3D objects in the scene, and $E = (o_i, r_{ij}, o_j)$ represents the directed relations r_{ij} between object pairs. Rather than training a dedicated open-vocabulary 3D segmentation model on point clouds, we use 2D segmentation [17] and detection models to extract object information from images. Specifically, a segmentation model is used to produce instance masks, and an open-vocabulary detector provides object labels. Given a sequence of RGB-D frames along with camera intrinsic and poses, we project 2D masked pixels into 3D via depth lifting. Points from multiple views are fused into unified object-level point clouds based on geometry similarity $\phi_{geo}(i, j) = nnratio(p_{t,i}, p_{o_j})$, which is the proportion of points in point cloud $p_{t,i}$ of object t in frame i that have nearest neighbors in point cloud p_{o_j}).

For relation prediction, we treat each object pair $(o_i, o_j) \in V \times V$ as a candidate edge and use our proposed SPOT to infer the relation. The predicted relation is then assigned back to the corresponding 3D object pair using object IDs and masks. After iterating through all available frames and pruning redundant or spurious edges, we obtain the final 3D scene graph that represents the semantic and spatial structure of the environment.

E.2. Duplicated Nodes Removal

The pipeline is capable of combining parts of objects into a unified representation by leveraging both spatial and semantic similarity across views; however, we acknowledge that in certain cases—particularly for large, planar structures like floors or walls—this fusion can be imperfect due to limitations in frame sampling or camera motion.

The framework leverages two types of similarity scores to match newly detected objects with previously stored ones, spatially and semantically:

- **Spatial IoU:** Computed based on the overlap between lifted partial point clouds. If the IoU > 0.5, we consider these two objects to be spatially aligned.

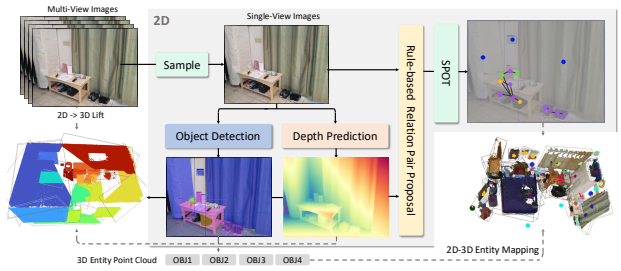

Figure 9. **Overview of the 2D-to-3D scene graph generation pipeline.** Given multi-view images, single-view frames are sampled for object detection and depth prediction. The predicted depth maps and object boxes are used to lift pixels into 3D entity point clouds. Concurrently, the 2D object boxes are utilized to generate rule-based object pair proposals, which are then passed to the SPOT to predict visual relations. Finally, the predicted 2D relations are mapped back into 3D using the pre-computed entity correspondence, resulting in a grounded 3D scene graph.

- **Semantic Similarity:** A contrastive score computed from CLIP embeddings extracted from object labels, defined as:

$$\Phi_{\text{sem}}(i,j) = \frac{f_i^T f_j}{2} + \frac{1}{2},$$

where f_i and f_j are the CLIP embeddings for object i and j. If the score > 0.5, we believe they belong to the same object category.

These two scores are jointly used to fuse identical objects observed from multiple views and to remove duplicates. In our experiments, we observe that small objects' point clouds can typically be fused and recovered reliably from a few frames. For large objects (e.g., floors, walls, ceilings), the fusion is sensitive to two factors:

- **Frame sampling frequency:** Sparse sampling may increase disparity between views, reducing overlap and similarity.
- **Camera motion:** Adjacent frames may have limited field-of-view overlap.

One feasible method to mitigate this issue is to increase the sampling rate to increase the adjacent overlap. However, when camera movement is too large, this problem still exists, which is a limitation of most existing methods that follow this frame-wise approach by lifting 2D detections to 3D. This is an exciting direction for future exploration.

E.3. Edge Cases in 3DSG

Since the 3D scene graph is generated from images of the scene and relies on generating 2D scene graphs as the "intermediate" step, there is a limited corner case when two objects have never been co-observed in any images, but have some semantic relation between them. This limitation is common to many current 3D scene graph generation [10] approaches that rely on 2D, per-frame observations. Like prior approaches, our method assumes that meaningful object relations emerge from co-visible object pairs within at least one view. Handling disjoint objects from sparse views remains an open challenge.

While our pipeline theoretically could predict relations for such disjoint object pairs if global relation proposals are provided, as mentioned in the question, these predictions would rely more heavily on textual priors instead of visual cues. We see this scenario as a valuable direction for future work: developing a multi-view context model allowing global relation queries across frames to support relation inference in sparse-view settings.

F. More Qualitative Results

In this section, we provide more qualitative results on different datasets. In Fig. 10, Fig. 11, Fig. 12, we separately display more visualizations on three cross-domain datasets: PSG, 3DSSG, and ScanNet [8], showing the generalization ability of our model. In Fig. 13, we also present results on PSG that illustrate how SPOT predicts richer spatial relations (e.g. "behind"), even though the ground-truth answers are different. This finding raises the need for a more robust and comprehensive evaluation method to quantify different perspectives of the scene graph prediction performance.

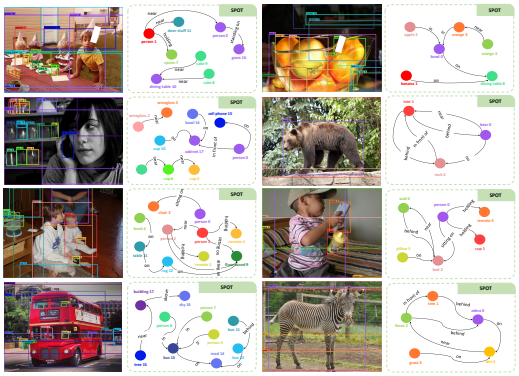

Figure 10. Additional cross-domain qualitative results on Panoptic Scene Graph Dataset. We visualize only the relations that exist in the ground truth.

G. Limitations

While SPOT demonstrates strong generalization by leveraging a vision-language model (VLM), it remains inherently constrained by the prior knowledge and reasoning abilities of the underlying language model. This reliance limits its ability to handle relationships or concepts that fall too different from the pretrained distribution. In terms of efficiency, SPOT trades off throughput for generalization. Unlike traditional approaches such as OvSGTR [6], which utilize lightweight architectures like Grounding DINO and simple MLP classifiers, our VLM-based method incurs additional

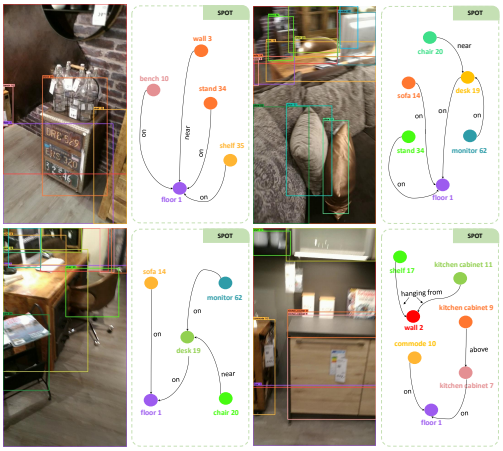

Figure 11. Additional cross-domain qualitative results on 3DSSG. We visualize only the relations that exist in the ground truth.

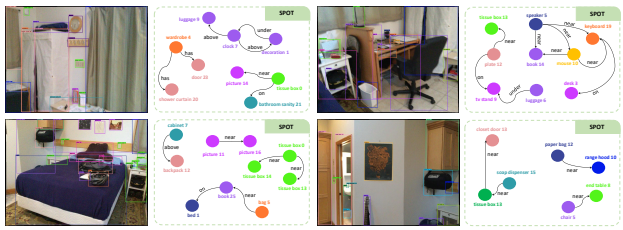

Figure 12. Qualitative cross-domain results on ScanNet. In contrast to PSG and 3DSSG, ScanNet is not a standard scene graph dataset. We include these results to show the generalization ability of SPOT and effectiveness in practical use with a real object detections.

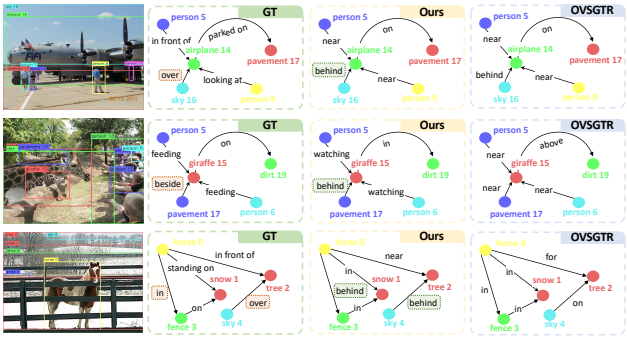

Figure 13. Additional cross-domain qualitative results on Panoptic Scene Graph Dataset. We observe that SPOT predicts richer spatial relations, which we highlight with green boxes.

computational overhead. Conventional methods can predict thousands of relation pairs within seconds, whereas prompting a large language model for the same quantity of relations introduces latency and resource demands. Another potential limitation arises from the length of the prediction sequence. As the number of predicted triplets grows, the prompt length increases accordingly, which can lead to degraded performance due to the model's limited context window. Although we attempt to mitigate this by segmenting long prompts into smaller batches during inference, this issue of forgetting remains a broader challenge in both the scene graph generation downstream task and general language modeling.

H. Social impact & Safeguards

A key application of 3D scene graph models is their integration into embodied agent systems for planning and interaction, where agents rely on spatial relation triplets and object locations to make decisions. Prediction of scene graphs for this purpose offers the potential for more explainable decision-making in these critical systems. However, current quantitative evaluation protocols for open-world 3D scene graph generation remain limited. To explore the full value of these systems to improve the safety of robotic applications, further quantitative vetting becomes essential to ensure that both relationship predictions and object detections faithfully represent the physical environment. Without such safeguards, erroneous predictions could result in agents' interaction with objects in unsafe or unintended ways, posing potential risks in real-world deployment.