FunFact: Building Probabilistic Functional 3D Scene Graphs via Factor-Graph Reasoning

Zhengyu Fu[1] René Zurbrügg[1] Kaixian Qu[1] Marc Pollefeys[1,2]
Marco Hutter[1] Hermann Blum[3†] Zuria Bauer[1†]

[1]ETH Zürich [2]Microsoft [3]University of Bonn & Lamarr Institute
[†]Equal supervision

Abstract

Recent work in 3D scene understanding is moving beyond purely spatial analysis toward functional understanding, such as discovering which knob controls which burner or which switch toggles which light. Existing methods, however, reason over each functional relation independently, missing the scene-wide interdependencies that humans use to resolve ambiguity: if one knob controls a burner, the others become less likely candidates to control the same burner. We introduce FunFact, a framework for constructing probabilistic open-vocabulary functional 3D scene graphs from posed RGB-D images. FunFact first builds an object- and part-centric 3D map and uses foundation models to propose plausible functional-relation templates. Candidate functional edges are then instantiated from these templates and encoded as binary variables in a dual factor graph, where inter-edge dependencies are inferred from visual-geometric evidence and enforced through higher-order factors. To benchmark FunFact, we introduce FunThor, a synthetic dataset with exhaustive functional annotations across 12 AI2-THOR scenes. Experiments on existing real-world datasets and FunThor demonstrate that FunFact significantly improves functional-relation recall and reduces calibration error for ambiguous relations, highlighting the benefits of holistic probabilistic modeling for functional scene understanding.

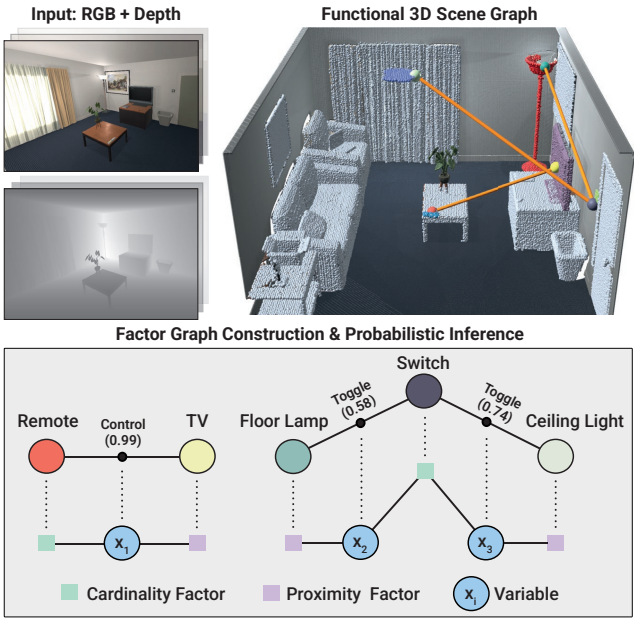

Figure 1. ***FunFact* for functional scene understanding.** Given posed RGB-D inputs, *FunFact* reconstructs an object- and part-centric 3D map and builds a functional scene graph (top). Candidate relations are encoded as binary variables in a dual factor graph (bottom), where cardinality and proximity factors jointly resolve ambiguities via belief propagation, yielding calibrated per-edge confidence scores.

1. Introduction

Functional scene understanding—capturing *how* entities interact rather than merely *what* and *where* they are—is increasingly recognized as a key frontier in computer vision [2, 10, 12]. Such information is crucial for embodied agents that must reason about acting in everyday environments [5, 6].

Many functional relations are not fully observable from static vision alone: a light switch and its lamp may be far apart or occluded, and multiple stove knobs may be spatially indistinguishable. This calls for models that reason jointly over all candidate relations and produce confidence scores that reflect scene-wide context, rather than independent pairwise decisions.

Related work. Open-vocabulary 3D scene graph methods [3, 11] capture semantic and spatial structure but leave functional interactions unaddressed. Early functional scene understanding benchmarks such as SceneFun3D [2] focus on part-level affordances with limited inter-object interaction annotations. OpenFunGraph [12] and FunGraph [10]

propose open-vocabulary functional 3D scene graphs using LLMs and 2D VLMs, but treat each edge independently. *FunFact* fills this gap by introducing a probabilistic, factor-graph-based formulation that jointly reasons over all functional edges in the scene.

This work. We propose *FunFact*, which builds open-vocabulary functional 3D scene graphs from posed RGB-D images and refines all candidate functional edges jointly via dual factor graph inference. Candidate relations become binary variables; LLM-derived structural priors and geometric proximity cues are encoded as factors. Belief propagation then yields per-edge confidence scores that reflect scene-wide context. Our main contributions are:

- A robust pipeline for reconstructing open-vocabulary functional 3D scene graphs from posed RGB-D inputs.
- A factor-graph formulation that jointly infers all functional relations and produces better-calibrated per-edge confidence estimates.
- A new synthetic benchmark (*FunThor*) with exhaustive functional annotations enabling precision, recall, and calibration evaluation.

2. Method

FunFact consists of two stages: (i) object- and part-centric scene reconstruction, and (ii) functional scene graph creation via a dual factor graph, as illustrated in Fig. 2.

2.1. Scene Reconstruction

Given posed RGB-D observations, we query a VLM (`GPT-4.1`) per frame to propose functional objects, their parts, and coarse bounding boxes. GroundingDINO [9] grounds these proposals, cross-validated against the VLM bounding boxes to suppress hallucinations. A second GroundingDINO pass within each object crop localizes fine-grained functional parts, which are filtered by geometric consistency. All detections are back-projected into 3D and fused across views following BBQ [8], yielding an object- and part-centric 3D map that serves as the backbone for downstream factor graph inference.

2.2. Functional Scene-Graph Creation

Given the 3D map, *FunFact* constructs candidate functional relations and encodes them as binary variables in a dual factor graph (Fig. 3). The "dual" nature stems from inverting the scene graph structure: scene graph nodes become constraint factors, and scene graph edges (candidate relations) become the variables to be inferred.

LLM-based functional relation priors. For every object with functional parts, we query an LLM with the object's label, description, and part labels, asking it to propose plausible semantic functional relation templates (e.g., *knob controls burner*). For each relation type, the LLM also predicts whether it is typically *one-to-one* or follows a more flexible cardinality pattern. We represent proposed relation templates for the k-th object as $\mathcal{T}_k = \{r_{k,j}\}_{j=1}^{M_k}$, where each $r_{k,j}$ specifies the semantic types of the two endpoints and the relation predicate; M_k is the number of proposed templates for the k-th object.

Dual factor graph construction. For each relation template $r_{k,j}$, we enumerate all part–object and part–part combinations matching the template's semantic types and connect them as candidate edges $\mathcal{E}_{k,j} = \{e_i^{k,j}\}_{i=1}^{E_{k,j}}$ (e.g., a complete bipartite graph between all knobs and burners on a stove), where $E_{k,j}$ is the total number of edges resulting from this exhaustive match. From these edges, we construct a local factor graph with variables $\mathcal{X}_{k,j} = \{x_i^{k,j}\}_{i=1}^{E_{k,j}}$. Each binary variable $x_i^{k,j} \in \{0,1\}$ is the dual of edge $e_i^{k,j}$, indicating whether that functional edge is present.

Cardinality-based constraint factors. To encode structural priors such as one-to-one mappings, we introduce *cardinality factors* $\phi_{\text{card}}(\cdot)$ that penalize configurations where a single part connects to multiple counterparts or to none. Concretely, for a part node n involved in one-to-one relations (e.g., a specific knob), let $\mathcal{X}_n$ denote the variables whose dual edges are incident to n, and let $d_n = \sum_{x \in \mathcal{X}_n} x$ be the number of active connections. We define the cardinality factor as:

$$\phi_{\text{card}}(\mathcal{X}_n) = \begin{cases} b^{d_n - 1} & \text{if } d_n \geq 1, \\ b^2 & \text{if } d_n = 0, \end{cases}$$

where $b \in (0,1)$ controls penalty strength, favoring structurally plausible one-to-one assignments.

Proximity-based prior factors. Each variable $x_i^{k,j}$ receives a unary proximity prior based on the 3D distance of its dual edge:

$$\phi_{\text{prox}}(x_i^{k,j}) = e^{-d(e_i^{k,j})/\lambda_{k,j}}, \tag{1}$$

where $d(e_i^{k,j})$ is the 3D distance of the edge $e_i^{k,j}$ and $\lambda_{k,j}$ is a scaling parameter defined as the median length of all edges in the local candidate edge set $\mathcal{E}_{k,j}$. This biases the model toward closer connections while still allowing cardinality factors to override proximity when necessary (e.g., a switch is less likely connected to a lamp with active connections to other switches, even if the switch is closer to the lamp).

Object–Object functional proposal. Analogously, the LLM proposes inter-object functional relations (e.g., *sponge cleans countertop*) with cardinality patterns. However, for object–object relations, we do not assume the proximity prior by default, but instead instruct the LLM to suggest which relations require proximity (e.g., curtains cover windows). For relations marked as one-to-one or proximity-sensitive, or both, we instantiate local factor graphs with the same cardinality and proximity factors, jointly optimized with part-centric edges in a global dual factor graph.

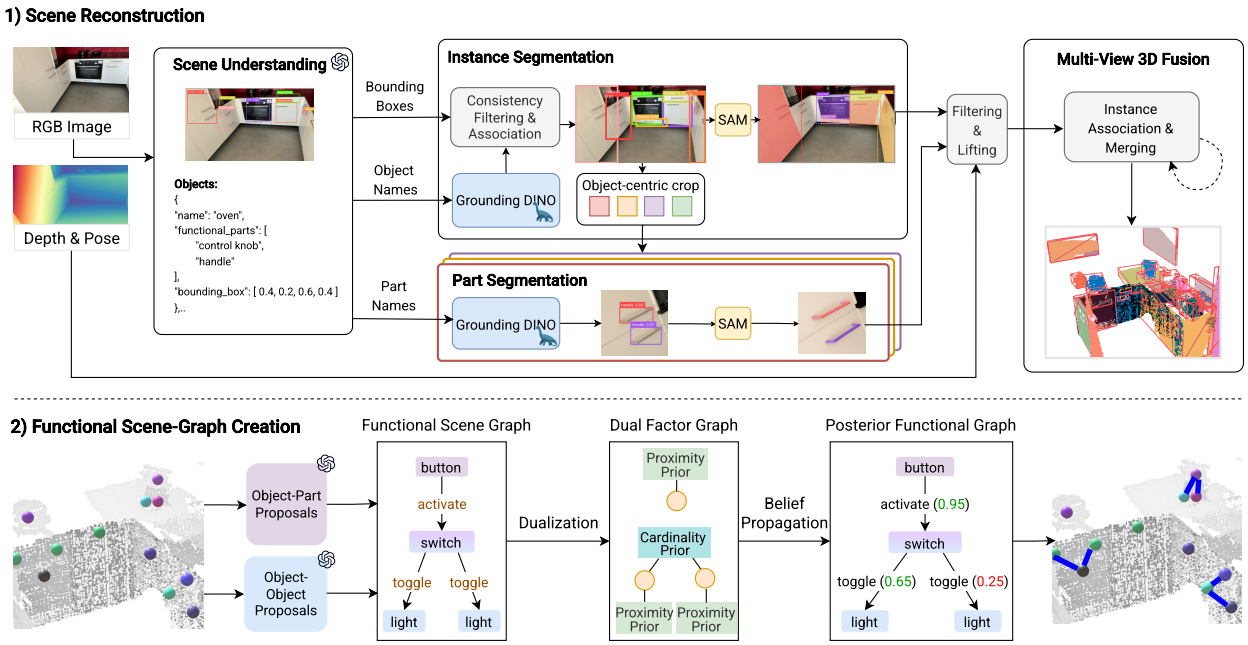

Figure 2. **Overview of *FunFact*.** Given posed RGB-D images, *FunFact* builds a functional 3D scene graph in two stages. *(i) Scene Reconstruction:* a VLM proposes functional objects, their part labels, and coarse 2D bounding boxes; GroundingDINO and SAM ground these into instance masks, which are cross-validated against the VLM bounding boxes to suppress hallucinations. Multi-view fusion lifts the detections into 3D and aggregates them across frames, yielding a part-aware 3D map. *(ii) Functional Scene Graph Creation:* an LLM proposes object-object and object-part relation templates; candidate relations are instantiated and encoded as binary variables in a dual factor graph; cardinality and proximity factors are resolved via belief propagation to yield per-edge confidence scores.

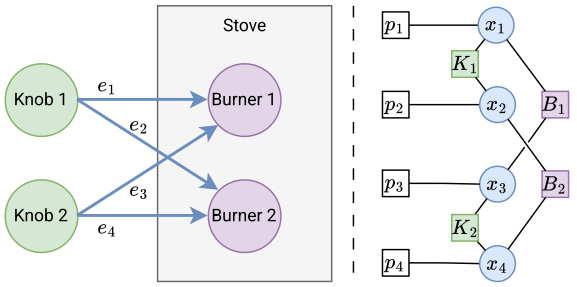

Figure 3. **Functional scene graph and its dual factor graph.** *Left:* Candidate functional scene graph with edges $e_1, \ldots, e_4$ representing knob–burner relations. *Right:* The dual factor graph, where each binary variable x_i ◯ is the dual of edge e_i. □ p_i: unary proximity prior; □ K_i: cardinality factor per knob; □ B_i: cardinality factor per burner.

Probabilistic inference. We implement the dual functional factor graph using pgmpy [1] and perform belief propagation to infer the joint distribution over all candidate functional edges. To accelerate inference, we identify disjoint connected components, denoted as $\mathcal{C}_m (m = 1, 2, ..., M)$, which are isolated subgraphs that do not share prior or constraint factors (e.g., knowing a knob controls a burner does not help disambiguate connections between remote controls and TVs), and run inference on each component separately. For a given component $\mathcal{C}_m$ with variable set $\mathcal{X}_m$, the joint distribution is $P(\mathcal{X}_m) = \frac{1}{Z_m} \prod_{x \in \mathcal{X}_m} \phi_{\text{prox}}(x) \prod_{f \in \mathcal{F}_m} \phi_{\text{card}}(\partial f)$, where $\mathcal{F}_m$ denotes the cardinality factors in $\mathcal{C}_m$, $\partial f \subseteq \mathcal{X}_m$ the variables connected to factor f, and Z_m a normalization constant. After convergence, we marginalize this distribution to obtain per-edge confidence scores, which are thresholded to produce the final functional scene graph.

3. Results

We evaluate *FunFact* on FunGraph3D [12] and our newly introduced *FunThor* benchmark, assessing mapping quality, functional relation recall, and confidence calibration.

3.1. Functional AI2-THOR (*FunThor*)

Existing datasets for functional scene understanding are limited in annotation density: SceneFun3D focuses on part-level affordances, while FunGraph3D extends to inter-object relations but has sparse, partially heuristic annotations. We introduce *Functional AI2-THOR (FunThor)*, a synthetic benchmark built on AI2-THOR [7] with 12 scenes spanning 4 environment types (kitchen, living room, bedroom, bathroom), 26 types of functional relations, and 720

Table 1. **Mapping and triplet evaluation on FunGraph3D.** Recall@K for objects and interactive elements (Inter. Elem.), and overall triplet recall. Higher is better; bold: best per column.

	Objects (↑)		Inter. Elem. (↑)		Triplets (↑)	
Methods	R@3	R@10	R@3	R@10	R@5	R@10
OpenFunGraph	70.7	79.1	44.4	57.6	29.8	45.0
FunFact (Ours)	**91.1**	**96.6**	**68.3**	**78.7**	**48.7**	**63.9**

Table 2. **Functional edge evaluation on *FunThor*.** We report Precision, Recall, and F1 (higher is better) for all predicted functional edges, and ECE (lower is better) on the subset for which OpenFunGraph provides confidence scores.

Methods	Prec. [%] (↑)	Recall [%] (↑)	F1 [%] (↑)	ECE (↓)
OpenFunGraph	23.4	12.2	16.0	0.25
FunFact (Ours)	**31.9**	**49.3**	**38.7**	**0.11**

posed RGB-D frames. All annotations are automatically generated from the simulator's native object properties and affordances, ensuring comprehensive coverage.

3.2. Mapping and Triplet Evaluation

We follow the evaluation protocol of OpenFunGraph [12], measuring Recall@K for reconstructed functional nodes (objects and parts) and predicted subject–predicate–object triplets. As shown in Tab. 1, *FunFact* substantially outperforms OpenFunGraph on FunGraph3D across both mapping and triplet metrics. The largest mapping gains are on interactive elements (+23.9pp R@3), where our hierarchical object-part pipeline reliably detects fine-grained functional parts missed by the flat object-centric approach of OpenFunGraph. For triplet prediction, *FunFact* improves overall recall by 18.9pp at both R@5 and R@10, highlighting the effectiveness of its two-stage prediction pipeline for functional relation discovery.

3.3. Comprehensive Evaluation on *FunThor*

FunThor's dense annotations enable a comprehensive evaluation of precision, recall, F1, and expected calibration error (ECE) [4]. ECE measures the alignment between predicted confidence and empirical accuracy (lower is better). Since OpenFunGraph only provides confidence scores for edges connected to switches, outlets, and remote controls, we restrict the ECE comparison to this subset for a fair comparison. As shown in Tab. 2, *FunFact* outperforms OpenFunGraph across all metrics, with ECE dropping from 0.25 to 0.11. This improvement reflects the factor graph's ability to leverage scene-wide context and suppress overconfident predictions on ambiguous relations that pairwise methods cannot resolve.

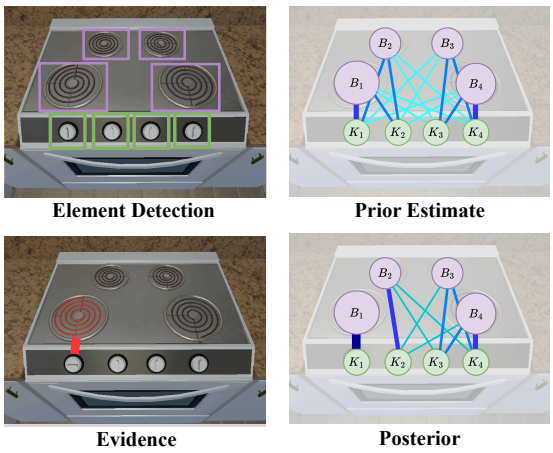

Figure 4. **Incorporating new evidence.** Our dual factor graph updates functional predictions as new evidence becomes available. Thicker and darker blue edges indicate higher confidence, while thinner and lighter edges represent lower-confidence associations. Initially, each knob-burner edge is scored using proximity and cardinality priors alone. After observing that the left knob controls the left burner, the model propagates this evidence to update the remaining three knob-burner associations.

4. Future Extensions

The dual factor graph is not restricted to the proximity and cardinality priors used here. Any inter-object or inter-edge structural information, such as object co-occurrence statistics and semantic compatibility scores, can be incorporated as additional unary or higher-order factors without modifying the inference procedure. This extensibility is especially compelling for evidence integration: once observations are injected into the factor graph, belief propagation globally updates all structurally related edges (Fig. 4).

5. Conclusion

We presented *FunFact*, a two-stage framework that reconstructs open-vocabulary functional 3D scene graphs from posed RGB-D images and jointly refines all candidate functional edges via dual factor graph inference. By encoding proximity and cardinality priors as higher-order factors, *FunFact* resolves scene-wide ambiguities that pairwise methods cannot address, yielding better-calibrated per-edge confidence scores. To enable detailed evaluation of precision and confidence, we introduce *FunThor*, a synthetic benchmark built on AI2-THOR [7] with more systematic and comprehensive functional annotations than existing datasets. Across FunGraph3D [12] and *FunThor*, *FunFact* consistently outperforms state-of-the-art baselines, validating the benefits of holistic probabilistic modeling for functional scene understanding.

Acknowledgements

This work was partially supported by the ETH AI Center, the Swiss National Science Foundation through the National Centre of Competence in Digital Fabrication (NCCR dfab), and Huawei Tech R&D (U.K.) through a research funding agreement. Additional support was provided by ETH Foundation Project 2025-FS-352 and SNSF Advanced Grant 216260. The authors also thank Dr. Cesar Dario Cadena Lerma for his insightful feedback on the mathematical notation used in this work, and Wanru Zhao for her expert assistance in the preparation of the figures.

References

[1] Ankur Ankan and Johannes Textor. pgmpy: A python toolkit for bayesian networks. *Journal of Machine Learning Research*, 25(265):1–8, 2024. 3

[2] Alexandros Delitzas, Ayca Takmaz, Federico Tombari, Robert Sumner, Marc Pollefeys, and Francis Engelmann. SceneFun3D: Fine-Grained Functionality and Affordance Understanding in 3D Scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1

[3] Qiao Gu, Ali Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, et al. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5021–5028. IEEE, 2024. 1

[4] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017. 4

[5] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR, 2022. 1

[6] Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander T Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu, Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Nikhil J. Joshi, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. Do as i can, not as i say: Grounding language in robotic affordances. In *Proceedings of The 6th Conference on Robot Learning*, pages 287–318. PMLR, 2023. 1

[7] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017. 3, 4

[8] Sergey Linok, Tatiana Zemskova, Svetlana Ladanova, Roman Titkov, Dmitry Yudin, Maxim Monastyrny, and Aleksei Valenkov. Beyond bare queries: Open-vocabulary object grounding with 3d scene graph. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13582–13589. IEEE, 2025. 2

[9] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pages 38–55. Springer, 2024. 2

[10] Dennis Rotondi, Fabio Scaparro, Hermann Blum, and Kai O. Arras. Fungraph: Functionality aware 3d scene graphs for language-prompted scene interaction. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4083–4090, 2025. 1

[11] Abdelrhman Werby, Chenguang Huang, Martin Büchner, Abhinav Valada, and Wolfram Burgard. Hierarchical open-vocabulary 3d scene graphs for language-grounded robot navigation. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024. 1

[12] Chenyangguang Zhang, Alexandros Delitzas, Fangjinhua Wang, Ruida Zhang, Xiangyang Ji, Marc Pollefeys, and Francis Engelmann. Open-vocabulary functional 3d scene graphs for real-world indoor spaces. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 19401–19413, 2025. 1, 3, 4