

CHAD-SLAM: Charting CAD Models in Visual SLAM

Saad Ejaz¹, Ali Tourani¹, Miguel Fernandez-Cortizas¹, Javier Civera², Holger Voos¹, and Jose Luis Sanchez-Lopez¹

Abstract—CAD-based scene representations provide compact, structured geometric and semantic maps that are useful for robotic interaction and digital twins. Existing CAD-based reconstruction methods, however, typically operate offline and require precomputed camera poses. In contrast, we present CHAD-SLAM, a monocular visual SLAM system that constructs and maintains a CAD-based scene representation online. CHAD-SLAM extends a sparse visual SLAM backbone with CAD-to-image alignment and planar layout estimation and jointly optimizes camera poses, object states, and building components (walls, floor, and ceiling) within a joint bundle-adjustment. Furthermore, vision foundation models enable operation without scene-specific tuning, thereby enabling zero-shot indoor operation. Experiments on ScanNet show improved camera trajectory accuracy over the underlying visual SLAM baseline while simultaneously producing reliable CAD-based maps.

I. INTRODUCTION

Modeling real-world indoor scenes using CAD-based representations has gained significant attention for its ability to provide structured, semantically rich environment representations for robotics, simulation, and digital twin applications [1], [2]. Existing methods typically process videos or 3D scans offline, detecting objects, obtaining CAD models, and globally optimizing their poses to obtain a consistent scene representation [3], [4]. Although these methods produce high-quality reconstructions, their offline processing and substantial computational cost make them unsuitable for robots that require scene representations to be constructed and updated online.

CAD-based maps are well suited to this setting because they represent scenes as structured collections of objects rather than dense geometry. Compared to point clouds, meshes, or neural scene representations, repeated object instances can be stored as shared CAD prototypes paired with their pose and scale (9-DoF object state), with substantially lower memory overhead while preserving the geometric and semantic information needed for scene understanding and interaction [2]. These challenges motivate a visual Simultaneous Localization and Mapping (SLAM) system capable of incrementally constructing and maintaining a coherent CAD-based scene model.

To this end, we propose **CHAD-SLAM (Charting CAD models in visual SLAM)**, a monocular visual SLAM system that incrementally builds a CAD-based map within a sparse SLAM framework. CHAD-SLAM couples a visual odometry frontend with a CAD-to-image fitting module to estimate object states while incorporating planar *building components* [5],

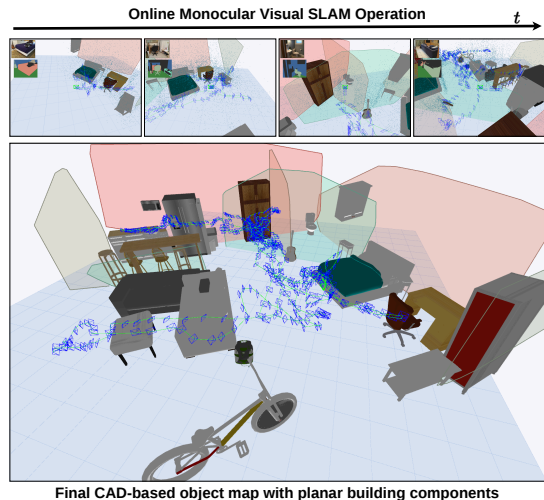


Fig. 1: CHAD-SLAM operates online from RGB images, fitting CAD models to segmented objects in keyframes and jointly optimizing them with camera poses and planar building components to produce a consistent indoor object-level map. The blue plane represents floor; upright polygons denote walls.

including walls, floors, and ceilings, as explicit map primitives. Its backend jointly optimizes camera poses, object states, and building components through object- and layout-aware factors. The semantic components operate without scene-specific tuning, enabling zero-shot generation of CAD-based *indoor* scene maps with explicit layout structure, as shown in Figure 1.

II. RELATED WORK

A. Object-level Visual SLAM

Object-level SLAM extends geometry-only maps with semantic object representations, enabling higher-level scene understanding. Early works use compact primitives such as cuboids [6] and quadrics [7], [8], which are efficient but discard fine shape information. Later methods adopt implicit representations such as SDFs [9] and NeRFs [10], which improve reconstruction quality but remain limited to observed surfaces. To address this, learned shape priors such as variational autoencoders in NodeSLAM [11] and DeepSDF embeddings in DSP-SLAM [12] and UncertainShapePose [9] enable complete reconstructions of unseen instances, though at the cost of per-category training and RGB-D input. Hydra++ [13] integrates complete shape estimation into an online hierarchical scene graph but similarly relies on depth sensors and is primarily a mapping method. SLAM++ [14] uses pre-scanned CAD models for complete geometry by construction, but requires a database of exact object instances existing in the environment to be known in advance. Instead, CHAD-SLAM combines online monocular SLAM with zero-shot indoor CAD-to-image alignment, producing complete-object

¹Authors are with the Automation and Robotics Research Group, Interdisciplinary Centre for Security, Reliability, and Trust (SnT), University of Luxembourg, Luxembourg. Holger Voos is also associated with the Faculty of Science, Technology, and Medicine, University of Luxembourg, Luxembourg. {saad.ejaz, ali.tourani, miguel.fernandez, holger.voos, joseluis.sanchezlopez}@uni.lu

²Author is with I3A, Universidad de Zaragoza. jcivera@unizar.es

This research was funded, in whole or in part, by the Luxembourg National Research Fund (FNR) under the DEUS Project (Ref. C22/IS/17387634/DEUS).

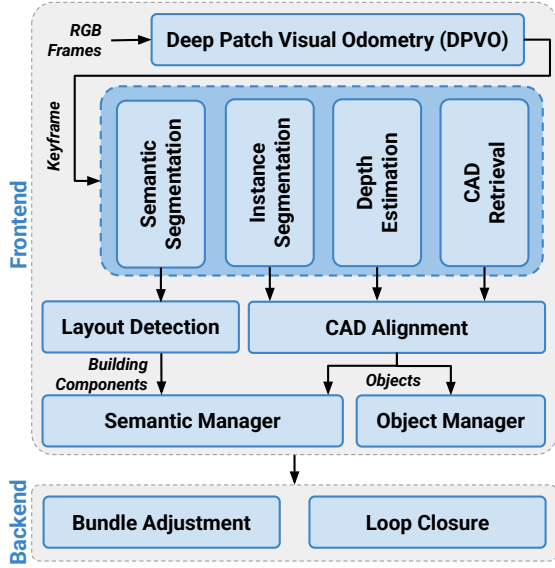


Fig. 2: Overall architecture of CHAD-SLAM. The frontend tracks visual patches and asynchronously generates object and layout observations. The semantic and object managers maintain persistent map entities, which are jointly optimized with camera poses in the backend.

maps without depth sensors, per-category training, or prior environment knowledge.

B. CAD-based Scene Reconstruction

CAD-based reconstruction methods decompose a scene into object instances represented by CAD models. Scan2CAD [2] and SceneCAD [15] introduced supervised CAD retrieval and alignment from RGB-D scans, establishing the standard benchmarks for this task. Methods such as POSITION [16] extend this to open-set CAD databases but still require complete 3D point clouds as input. Single-image methods such as ROCA [17] and SPARC [18] extend this to monocular input, while zero-shot approaches, such as Diorama [19], ZeroCAD [20], and SUFLECA [21], remove the need for pose annotations. Video-based methods such as Vid2CAD [3], FastCAD [4], InstaScene [22], and SimRecon [23] improve consistency across frames but operate offline over full sequences and require pre-computed camera poses. ShapeR [24] generates complete object meshes from SLAM-tracked sequences, but remains an offline mapping method. CHAD-SLAM addresses these limitations by jointly estimating camera poses, CAD-based object maps, and planar layout in an online manner.

III. PROPOSED METHOD

CHAD-SLAM builds upon DPV-SLAM [25], a monocular visual SLAM system that extends the DPVO [26] visual odometry. DPV-SLAM maintains patch-based visual features, refined through a bundle adjustment backend, which we extend to include objects and building components, as can be seen in Figure 2. The frontend detects and manages objects and building components, as described in §III-A, while the backend triggers and solves bundle adjustment following keyframe insertion and loop closures, as described in §III-B.

A. Frontend

The frontend consists of the DPVO *tracking* module, which produces visual patches tracked across input frames and governs keyframe selection; *CAD alignment*, which retrieves and aligns CAD models to keyframe images; *layout detection*, which detects planar building components; the *object manager*, which tracks and aggregates object observations across frames; and the *semantic manager*, which filters and re-associates all semantic map elements to ensure consistency over the lifetime of the map. CAD alignment and layout detection run at every confirmed keyframe; the object manager subscribes to every successful CAD alignment; and the semantic manager runs at fixed intervals.

1) *Layout Detection*: CHAD-SLAM models three planar building-component classes: walls, floor and ceiling, to constrain camera and object poses. Layout detection follows [5], applying semantic segmentation to each keyframe image and assigning DPVO patches to their respective category if their centers fall within a detected plane region. For each category and keyframe, a plane is fitted to the associated backprojected points using RANSAC, before being passed to the backend for joint optimization.

2) *CAD Alignment*: CAD alignment follows the pipeline of SUFLECA [21], though the system is agnostic to the choice of fitting method. Open-vocabulary detection and instance segmentation are first applied to localize object bounding boxes and extract masks. Monocular depth and CAD retrieval are then computed and passed, together with the object masks, to SUFLECA, which estimates a 9-DoF CAD-to-image transformation alongside a fit quality score $\mathcal{S}_{\text{fit}}$. The resulting alignments are aggregated by the object manager and used as observations in the backend for joint optimization.

3) *Object Manager*: The object manager maintains mapped objects by tracking DPVO patches assigned to instance masks, with object associations following [8]. Since CAD-to-image fits are noisy under motion blur and occlusion, fits of the same tracked object are clustered by translation, rotation, and scale using the NMS procedure of [3]. The cluster with the highest $\mathcal{S}_{\text{fit}}$ is kept active. Its highest-scoring fit determines the CAD model, while all fits in the active cluster contribute observations to bundle adjustment.

4) *Semantic Manager*: As all semantic elements are jointly optimized with camera poses, noise can adversely affect camera pose estimation. To mitigate this, the semantic manager applies several consistency checks. Objects with significant overlap with nearby instances are removed, and floor and ceiling observations deviating beyond a height threshold are discarded. This is typically caused by segmentation masks leaking onto flat surfaces such as tables and cabinets. Finally, mapped objects or building components that converge to similar poses after optimization are identified as re-association candidates and merged into a single map element.

B. Backend

The backend extends DPV-SLAM’s bundle adjustment by jointly optimizing camera poses, objects, and building components, with details that follow.

1) *State*: Let $\mathbf{T}_{wc}^{(i)} \in \text{SE}(3)$ denote the world-from-camera pose of keyframe i . Each mapped object o is represented by

$$\mathbf{x}_o^{(o)} = \left(\mathbf{T}_{wo}^{(o)}, \mathbf{s}_o^{(o)} \right), \quad (1)$$

where $\mathbf{T}_{wo}^{(o)} \in \text{SE}(3)$ is its world pose and $\mathbf{s}_o^{(o)} \in \mathbb{R}_+^3$ its anisotropic scale, yielding a 9-DoF object state. Building components are represented as planes following [5].

Camera and object translations are maintained in the arbitrary monocular SLAM gauge. A global scale $S > 0$ maps this gauge to metric units, while per-keyframe log-scale variables $s_f^{(i)}$ account for the error from the monocular metric depth estimator which is conditioned on single images. The effective metric scale at keyframe i is then

$$\lambda_i = S \exp\left(s_f^{(i)}\right). \quad (2)$$

Metric translations are recovered by applying the corresponding scale to their monocular-gauge estimates.

2) *Global scale estimation*: For an object observation (i, o) , let $\hat{\mathbf{t}}_{co}^{(i,o)}$ denote the metric translation provided by CAD alignment and $\tilde{\mathbf{t}}_{co}^{(i,o)}$ the corresponding translation predicted in the monocular SLAM gauge. Global scale is estimated as

$$S^* = \arg \min_{S>0} \sum_{(i,o) \in \mathcal{M}_S} \rho_H \left(\left\| S \tilde{\mathbf{t}}_{co}^{(i,o)} - \hat{\mathbf{t}}_{co}^{(i,o)} \right\|_2^2 \right), \quad (3)$$

where $\rho_H(\cdot)$ denotes the Huber loss and $\mathcal{M}_S$ contains a fixed number of observations sampled uniformly per object. The resulting S^* establishes the current global metric scale of the monocular reconstruction.

3) *Joint optimization*: Let $\mathcal{X}$ collect the active camera poses, object states, building components, and local scale variables. The backend minimizes

$$\begin{aligned} \mathcal{X}^* = \arg \min_{\mathcal{X}} & E_{\text{DPV}} + E_{\text{co}} + E_{\text{cp}} \\ & + E_{\text{fo}} + E_{\text{po}} + E_{\text{scale}}, \end{aligned} \quad (4)$$

where each term denotes a sum of information-weighted residuals robustified with a Huber loss. E_{DPV} contains the keyframe-pose and patch factors inherited from DPV-SLAM. The *camera-object term* E_{co} uses the metric CAD-to-camera observations produced by SUFLECA [21]. Each observation provides $\hat{\mathbf{T}}_{co}^{(i,o)}$, an object scale estimate, and a 9×9 information matrix over translation, rotation, and scale. The factor constrains $\mathbf{T}_{wc}^{(i)}$, $\mathbf{T}_{wo}^{(o)}$, $\mathbf{s}_o^{(o)}$, and $s_f^{(i)}$, with translations mapped from the monocular gauge to metric units through λ_i . The *camera-plane term* E_{cp} follows [5], using local RANSAC-based plane observations to jointly constrain camera poses and building components across keyframes. The *floor-object term* E_{fo} imposes a support constraint for object categories expected to rest on the floor, such as chairs, sofas, and tables. It penalizes the plane-to-plane discrepancy between the object's lower surface and the estimated floor plane, constraining both the object's translation along the gravity direction and the alignment of its lower-surface normal with the floor normal. The factor is applied only within a tolerance to avoid enforcing support in atypical configurations. The *patch-object term* E_{po} couples object-associated SLAM patches to the CAD geometry. Given observed patch centers $\mathcal{Q}_o$, transformed to

3D CAD space, and subsampled CAD surface points $\mathcal{C}_o$, we estimate

$$\Delta \mathbf{t}_o^* = \arg \min_{\Delta \mathbf{t}} \sum_{\mathbf{q} \in \mathcal{Q}_o} \rho_H \left(\min_{\mathbf{c} \in \mathcal{C}_o} \left\| \mathbf{q} - (\mathbf{c} + \Delta \mathbf{t}) \right\|_2^2 \right), \quad (5)$$

and use the resulting translation as a unary constraint on the object position. Finally, the *scale regularization term* E_{scale} penalizes the deviation of each keyframe's effective scale λ_i from the global scale S .

4) *Optimization schedule*: Bundle adjustment is performed over a sliding window of the most recent keyframes, while older connected keyframes remain fixed as anchors. This joint optimization runs asynchronously, and pose updates are accepted only when camera corrections remain below a prescribed threshold. The global metric scale is optimized more frequently than the full bundle adjustment because it is inexpensive to estimate and directly affects the metric semantic constraints. Loop closure is inherited from DPV-SLAM [25], after which the subsequent bundle adjustment is modified to include the objects and building components to maintain global consistency.

IV. EVALUATION

A. Implementation Details

We use GroundedSAM [27] for open-vocabulary object detection and segmentation, with a vocabulary associated with the CAD library of Scan2CAD [2], which suffices for indoor environments. Monocular depth is estimated using MoGe-2 [28], and CAD retrieval follows the pipeline of OSCAR [29], which was adapted by SUFLECA [21]. For camera tracking, we use the base variant of DPVO with the default configuration and a frame stride of 2. Bundle adjustment is triggered every 15 keyframes with a window size of 30, while scale optimization is triggered every 5 keyframes. For CAD-to-image alignment, we use the base variant of SUFLECA [21].

B. Methodology

We evaluate CHAD-SLAM on ScanNet [30] in terms of camera tracking and object mapping. Tracking is reported using the standard ATE RMSE. Object states are evaluated against Scan2CAD [2] annotations after Umeyama alignment of the estimated and ground-truth trajectories. Following Vid2CAD [3], an object alignment is *accurate* when translation, rotation, and scale errors are jointly below 20cm, 20°, and 20%, respectively. Since this metric does not penalize spurious predictions, we also report *precision* to capture map reliability and compactness.

For camera tracking, we compare against DPV-SLAM [25], the baseline underlying CHAD-SLAM. We augment it with the same monocular depth estimates used by CHAD-SLAM to recover metric scale by performing a least squares on randomly sampled patch points across all keyframes, denoting this variant as DPV-SLAM[†]. Notably, DPV-SLAM runs in synchronous mode. For object alignment, we reimplement Vid2CAD [3], replacing its single-image alignment module with SUFLECA [21], denoting it as Vid2CAD[†]. It receives the same posed keyframes as DPV-SLAM[†] and the same CAD-to-image alignments available to CHAD-SLAM.

Scene ID	0000_00	0005_01	0011_01	0024_00	0033_00	0137_02	0249_00	0286_00	Mean
Time (s)	185.9	48.3	91.9	113.2	47.4	88.8	91.1	109.8	97.1
Length (m)	73.1	16.3	43.9	43.2	29.4	31.3	41.7	38.4	39.6
SE(3) alignment									
DPV-SLAM [†] [25]	11.77 ± 0.64	7.21 ± 0.04	17.95 ± 4.05	8.61 ± 0.05	12.40 ± 0.06	9.65 ± 0.08	34.49 ± 5.54	9.96 ± 3.17	14.01
CHAD-SLAM	8.58 ± 0.89	6.54 ± 0.27	17.63 ± 1.76	8.06 ± 0.35	12.95 ± 0.41	9.07 ± 0.25	25.68 ± 2.84	8.98 ± 0.08	12.19
Sim(3) alignment									
DPV-SLAM [25]	7.17 ± 0.52	6.95 ± 0.01	23.95 ± 9.94	8.14 ± 0.05	11.92 ± 0.05	7.45 ± 0.06	24.71 ± 3.26	8.38 ± 0.11	12.33
DPV-SLAM [†] [25]	6.55 ± 0.14	6.93 ± 0.06	17.83 ± 3.95	8.16 ± 0.10	11.92 ± 0.03	7.43 ± 0.09	24.65 ± 3.20	9.54 ± 3.01	11.63
CHAD-SLAM	6.37 ± 0.42	6.52 ± 0.28	16.80 ± 1.68	8.05 ± 0.35	11.94 ± 0.20	6.62 ± 0.18	22.37 ± 2.55	8.21 ± 0.08	10.86
# Mapped objects	18.8 ± 0.8	7.0 ± 0.0	15.7 ± 1.0	14.2 ± 0.8	16.0 ± 0.0	11.5 ± 0.5	28.7 ± 2.0	10.0 ± 0.0	–

TABLE I: ScanNet trajectory error (ATE RMSE, cm), averaged over six runs per sequence and reported as mean $\pm$ standard deviation, with lower being better. [†] denotes our depth-augmented DPV-SLAM variant, which uses the same monocular depth estimates as CHAD-SLAM to recover metric scale and is therefore evaluated under SE(3) alignment. DPV-SLAM denotes the original scale-ambiguous monocular system. Final non-causal trajectories are evaluated to match the DPV-SLAM setting.

Method	Accuracy	Precision	F1-score
SE(3) alignment			
Vid2CAD [†] [3]	37.9	27.8	31.6
CHAD-SLAM	38.0	45.2	40.8
Sim(3) alignment			
Vid2CAD [†] [3]	42.6	29.9	34.4
CHAD-SLAM	38.7	45.7	41.4

TABLE II: Object-mapping performance averaged over six runs per sequence under SE(3) and Sim(3) trajectory alignment. All metrics are in % with higher being better. Vid2CAD[†] reconstructs an average of 27.3 objects per sequence, compared with 15.2 for CHAD-SLAM.

C. Results

Under SE(3) alignment, CHAD-SLAM improves ATE over DPV-SLAM[†] by approximately 13.0%. The gain is most pronounced on larger, object-rich sequences: 0249_00 maps the most objects and shows the largest improvement, followed by 0000_00, which ranks second in both. Longer sequences also provide more opportunities to revisit the same objects, yielding additional observations that strengthen their geometric constraints during optimization. An exception is 0033_00, where performance degrades despite the large number of mapped objects. One possible explanation is that the combination of short sequence duration and a larger number of semantic observations leaves fewer opportunities for asynchronous updates. Moreover, under Sim(3) alignment, CHAD-SLAM improves ATE over DPV-SLAM[†] by 6.6%. The smaller gain is expected because similarity alignment absorbs part of the scale error corrected by CHAD-SLAM, which uses the more robust object observations to recover metric scale instead of the randomly sampled patches.

Concerning object mapping, CHAD-SLAM’s main advantage over Vid2CAD[†] lies in maintaining a cleaner object map through effective association, fusion, and removal of erroneous or duplicate objects. This yields F1-score improvements of 9.2 and 7.0 percentage points under SE(3) and Sim(3) alignments, respectively. In raw alignment accuracy, however, CHAD-SLAM slightly lags Vid2CAD[†] by 3.9 percentage points under Sim(3) alignment, which can be attributed to two factors. First,

Vid2CAD[†] generates substantially more object hypotheses rather than fusing them, increasing the likelihood of obtaining a correct alignment. Second, Vid2CAD[†] optimizes using final corrected keyframe poses, whereas object accuracy in CHAD-SLAM is governed by the causal trajectory, meaning later trajectory corrections may only propagate to previously mapped objects after a subsequent global bundle adjustment, which occurs infrequently in the evaluated ScanNet sequences.

V. LIMITATIONS

Mapping objects as CAD models requires a multi-stage pipeline that introduces substantial GPU computation. Although the system architecture supports asynchronous online CAD alignment, our single-GPU evaluation precomputes these alignments to avoid interference with the latency-critical camera tracking. Consequently, the reported experiments do not yet demonstrate end-to-end real-time execution of the full pipeline. Running the pipeline only at keyframe frequency alleviates much of this contention; nevertheless, separate GPUs remain preferable. Another limitation is the CAD-to-image alignment accuracy, i.e., current state-of-the-art methods achieve only around 40% accuracy [21], [31], requiring conservative filtering and, consequently, yielding sparse object constraints, particularly for uncommon, symmetric, or textureless objects. This sparsity limits both map diversity and the extent to which semantic observations can improve camera pose estimation and metric scale recovery.

VI. CONCLUSION

We presented CHAD-SLAM, a monocular visual SLAM framework that incrementally builds structured scene representations from CAD objects and planar building components. By integrating zero-shot indoor CAD-to-image alignment with layout detection into a sparse visual SLAM pipeline, CHAD-SLAM jointly refines camera poses, object states, and scene layout within a unified bundle-adjustment framework. Our experiments on ScanNet show that these semantic constraints can improve trajectory estimation while simultaneously producing reliable CAD-based maps. Future work will focus on efficient online execution and supporting high-level robotic interaction and manipulation.

REFERENCES

- [1] U. Klank, D. Pangercic, R. Rusu, and M. Beetz, “Real-time cad model matching for mobile manipulation and grasping,” in 9th IEEE-RAS International Conference on Humanoid Robots, 2010.
- [2] A. Avetisyan, M. Dahnert, A. Dai, M. Savva, A. X. Chang, and M. Niessner, “Scan2cad: Learning cad model alignment in rgb-d scans,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.
- [3] K.-K. Maninis, S. Popov, M. Niesner, and V. Ferrari, “Vid2CAD: CAD Model Alignment Using Multi-View Constraints From Videos,” IEEE Transactions on Pattern Analysis & Machine Intelligence, 2023.
- [4] F. Langer, J. Ju, G. Dikov, G. Reitmayr, and M. Ghafoorian, “Fastcad: Real-time cad retrieval and alignment from scans and videos,” in Proc. European Conference on Computer Vision, 2024.
- [5] A. Tourani, S. Ejaz, H. Bavle, M. Fernandez-Cortizas, D. Morilla-Cabello, J. L. Sanchez-Lopez, and H. Voos, “vs-graphs: Tightly coupling visual slam and 3d scene graphs exploiting hierarchical scene understanding,” IEEE Robotics and Automation Letters, 2026.
- [6] S. Yang and S. Scherer, “Cubeslam: Monocular 3-d object slam,” IEEE Transactions on Robotics, vol. 35, no. 4, pp. 925–938, 2019.
- [7] X. Han and L. Yang, “Sq-slam: Monocular semantic slam based on superquadric object representation,” Journal of Intelligent & Robotic Systems, vol. 109, no. 2, p. 29, 2023.
- [8] Y. Wu, Y. Zhang, D. Zhu, Z. Deng, W. Sun, X. Chen, and J. Zhang, “An object slam framework for association, mapping, and high-level tasks,” IEEE Transactions on Robotics, 2023.
- [9] Z. Liao, J. Yang, J. Qian, A. P. Schoellig, and S. L. Waslander, “Uncertainty-aware 3d object-level mapping with deep shape priors,” in The IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2024, pp. 4082–4089.
- [10] X. Han, H. Liu, Y. Ding, and L. Yang, “Ro-map: Real-time multi-object mapping with neural radiance fields,” IEEE Robotics and Automation Letters, 2023.
- [11] E. Sucar, K. Wada, and A. Davison, “Nodeslam: Neural object descriptors for multi-view shape reconstruction,” in The International Conference on 3D Vision (3DV). IEEE, 2020, pp. 949–958.
- [12] J. Wang, M. Rünz, and L. Agapito, “Dsp-slam: Object oriented slam with deep shape priors,” in The International Conference on 3D Vision (3DV). IEEE, 2021, pp. 1362–1371.
- [13] H. Lim, N. Hughes, X. Yu, R. Xu, Y. Chang, J. Shi, R. Talak, and L. Carlone, “Hydra++: Real-time hierarchical 3d scene graph construction with object-level shape estimation,” in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2026.
- [14] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, “Slam++: Simultaneous localisation and mapping at the level of objects,” in The IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2013, pp. 1352–1359.
- [15] A. Avetisyan, T. Khanova, C. Choy, D. Dash, A. Dai, and M. Nießner, “Scenecad: Predicting object alignments and layouts in rgb-d scans,” in The European Conference on Computer Vision (ECCV), August 2020.
- [16] R. Yang, H. Liu, Y. Chen, S. Ao, Y. Zhang, L. Wang, K. Xue, S. Zhou, and Y. Guo, “Position: Open world 3d scene cad recomposition,” IEEE Transactions on Image Processing, 2026.
- [17] C. Gümeli, A. Dai, and M. Nießner, “Roca: Robust cad model retrieval and alignment from a single image,” in Proc. IEEE/CVF conference on Computer Vision and Pattern Recognition, 2022.
- [18] F. Langer, G. Bae, I. Budvytis, and R. Cipolla, “Sparc: Sparse render-and-compare for cad model alignment in a single rgb image,” in Proc. British Machine Vision Conference, 2022.
- [19] Q. Wu, D. Iliash, D. Ritchie, M. Savva, and A. X. Chang, “Diorama: Unleashing zero-shot single-view 3d indoor scene modeling,” in Proc. IEEE/CVF International Conference on Computer Vision, 2025.
- [20] P. Arsomngern, S. Khwanmuang, M. Nießner, and S. Suwajanakorn, “Zero-shot inexact cad model alignment from a single image,” in Proc. IEEE International Conference on Computer Vision, 2025.
- [21] S. Ejaz, M. Fernandez-Cortizas, J. Civera, H. Voos, and J. L. Sanchez-Lopez, “Suffeca: Scaling up feature learning for cad-to-image alignment,” arXiv preprint arXiv:2607.15058, 2026.
- [22] Z. Yang, B. Yang, W. Dong, C. Cao, L. Cui, Y. Ma, Z. Cui, and H. Bao, “Instascene: Towards complete 3d instance decomposition and reconstruction from cluttered scenes,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2025, pp. 7771–7781.
- [23] C. Xia, K. Zhu, Z. Wang, F. Liu, Z. Zhang, and Y. Duan, “Simrecon: Simready compositional scene reconstruction from real videos,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2026.
- [24] Y. Siddiqui, D. Frost, S. Aroudj, A. Avetisyan, H. Howard-Jenkins, D. DeTone, P. Moulon, Q. Wu, Z. Li, J. Straub, R. Newcombe, and J. Engel, “Shaper: Robust conditional 3d shape generation from casual captures,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2026.
- [25] L. Lipson, Z. Teed, and J. Deng, “Deep Patch Visual SLAM,” in European Conference on Computer Vision, 2024.
- [26] Z. Teed, L. Lipson, and J. Deng, “Deep patch visual odometry,” Advances in Neural Information Processing Systems, 2023.
- [27] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan et al., “Grounded sam: Assembling open-world models for diverse visual tasks,” arXiv preprint arXiv:2401.14159, 2024.
- [28] R. Wang, S. Xu, Y. Dong, Y. Deng, J. Xiang, Z. Lv, G. Sun, X. Tong, and J. Yang, “Moge-2: Accurate monocular geometry with metric scale and sharp details,” in Advances in Neural Information Processing Systems, 2025.
- [29] T. Pulli, J.-B. Weibel, P. Hönig, M. Hirschmanner, M. Vincze, and A. Holzinger, “Oscar: Open-set cad retrieval from a language prompt and a single image,” arXiv preprint arXiv:2601.07333, 2026.
- [30] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, “ScanNet: Richly-annotated 3d reconstructions of indoor scenes,” in The IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2017, pp. 5828–5839.
- [31] F. Langer, R. Cipolla, and I. Budvytis, “Sparse multi-object render-and-compare,” in Proc. British Machine Vision Conference, 2023.