UniFunc3D: Unified Active Spatial-Temporal Grounding for 3D Affordance Segmentation

Jiaying Lin **Dan Xu***
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
csjylin@gmail.com danxu@cse.ust.hk

Abstract

Affordance segmentation in 3D scenes requires an agent to ground implicit natural-language instructions into precise masks of fine-grained interactive elements. Existing training-free methods typically rely on fragmented pipelines, which introduce visual blindness during task parsing and limit accuracy through single-scale spatial and temporal processing. We present UniFunc3D, a unified and training-free framework that treats the multimodal large language model as an active observer. By utilizing a unified MLLM backbone, UniFunc3D performs joint semantic-temporal-spatial reasoning to ground task decomposition in direct visual evidence. Our approach introduces active spatial-temporal grounding with a coarse-to-fine strategy. This allows the model to select correct video frames adaptively and focus on high-detail interactive parts while preserving the global context necessary for disambiguation. On SceneFun3D, our UniFunc3D achieves state-of-the-art performance, surpassing prior training-free methods by a large margin with a relative 59.9% mIoU improvement, and even outperforming training-based methods without any task-specific training. Code is available on our project page: `https://jiaying.link/unifunc3d`.

1 Introduction

For an embodied agent to operate effectively in human environments, it must look beyond simple object labels and understand affordances, the latent functional properties that enable interaction. While standard open-vocabulary 3D segmentation focuses on identifying what an object is (*e.g.*, "a cabinet"), affordance segmentation requires determining how to interact with it based on intent. For instance, given the command "turn on the ceiling light," an agent must infer that the target is a specific wall switch, even if the word "switch" is never mentioned. This task is inherently difficult because it requires a synergy of high-level world knowledge for language interpretation and fine-grained spatial perception for localizing components at a small scale.

Existing training-free methods (*e.g.*, Fun3DU [7]) typically relies on fragmented pipelines that suffer from a fundamental lack of active spatial-temporal reasoning. These approaches often begin with a "visually blind" reasoning stage where a text-only LLM (*i.e.*, LLaMA-3.1-9B [14]) to decompose the input text description into two kinds of objects: the contextual object (*i.e.*, the object that contains or is related to the functional object) and the functional object (*i.e.*, the ultimate object(s) or parts to segment). Such a fragmented pipeline means that if one of these objects is wrongly detected, then the final result will be wrong. However, as their first stage is text-only, their decomposition occurs without seeing the scene, leading to inaccurate contextual and functional identifications. For example, for the input `plug the device in the left socket behind the armchair`, Fun3DU may wrongly identify the `device` as the functional object while the correct one is `socket`. Second, they

*Corresponding Author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

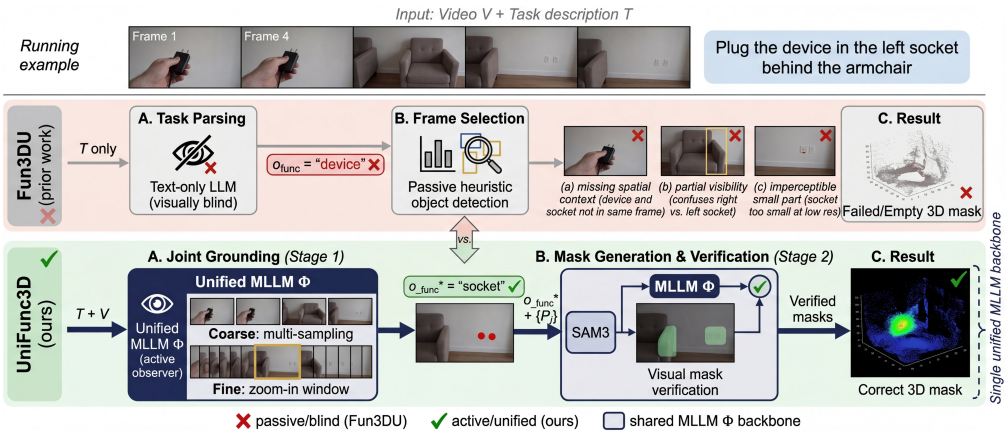


Figure 1: **Overview of UniFunc3D compared to existing fragmented pipelines.** (Top) Prior methods like Fun3DU rely on a visually blind text-only LLM for initial task parsing. Coupled with single-scale passive heuristic frame selection, this fragmented approach suffers from three critical failure modes: semantic misinterpretations (Task A→ Task B), spatial-temporal context inconsistencies (a, b) and imperceptible small target (c), leading to error in the final output. (Bottom) Our proposed UniFunc3D addresses these limitations by utilizing a unified MLLM as an active observer. By employing a coarse-to-fine active spatial-temporal grounding strategy alongside visual mask verification, UniFunc3D consolidates semantic, temporal, and spatial reasoning in a unified MLLM. This allows the model to accurately ground implicit targets and generate precise fine-grained 3D functional masks while preserving necessary global context.

utilize passive heuristic rules (*e.g.*, hand-crafted, threshold-based scores that weight the centeredness and uniformity of detected contextual object masks) to select video frames for independent processing. They identify all frames containing the contextual object based purely on object category and assume the detected contextual object and the target functional part always reside in the same frame, an ideal situation that is frequently not guaranteed. What is worse, these methods process images as isolated single frames and do not utilize temporal information from multiple frames, leading to errors when disambiguating spatial relationships or aggregating visibility across views. Third, when processing images, they rely on single-scale processing and lack a human-like "zoom-in" mechanism. Because it cannot adaptively focus on important frames at a higher resolution, small functional parts occupy tiny regions that appear as imperceptible noise.

We observe that these issues are not just isolated errors but are symptoms of a "perception-reasoning gap". Because existing methods cannot actively look for the context they need, they fall victim to cascading errors where a mistake in the initial "blind" reasoning or heuristic frame selection irrecoverably ruins all downstream steps. This motivates us to design a method that actively seeks out the necessary spatial-temporal context or aggregates multi-view context and multi-scale content to resolve fine-grained details for this challenging task.

To address these limitations, we introduce UniFunc3D, a unified and training-free framework that consolidates semantic, temporal, and spatial reasoning within a single multimodal large language model (MLLM). In UniFunc3D, we propose an active spatial-temporal grounding process that eliminates handcrafted and heuristic rules with the heavy hyperparameter controls and unstable object detection found in Fun3DU. It observes the entire video to adaptively identify informative temporal segments and enables the selection of optimal candidate frames based on direct visual evidence. During this process, the model jointly conducts semantic, temporal, and spatial reasoning to directly locate the target functional object among multiple input video frames. This unified architecture prevents cascading errors by allowing these distinct reasoning aspects to mutually inform and reinforce one another. This synergy is essential for resolving fine-grained interactive parts in complex 3D scenes. Besides, our UniFunc3D utilizes a coarse-to-fine strategy that mimics human-like perception: In the coarse round, the model surveys the video at low resolution to identify candidates. In the fine round, it processes a dense temporal window at native high resolution, resolving spatial anchors and small parts while retaining complete scene context for disambiguation. This coherent

architecture ensures the agent can self-correct its initial estimates, avoiding the error propagation inherent in naive zoom-in pipelines with region-cropping. Our key contributions include:

- Unified Multimodal Architecture: We eliminate the cascading errors of fragmented pipelines by consolidating reasoning and perception into a single, spatial-temporal and visual-aware MLLM.
- Active spatial-Temporal Grounding: We replace passive heuristics with a multi-sampling and verification strategy that allows the model to autonomously select the most informative content from video sequences.
- Human-like Coarse-to-fine Perception: Our two-round approach achieves high precision on fine-grained elements without external cropping, preserving global context for robust spatial reasoning.
- State-of-the-Art Performance: UniFunc3D achieves state-of-the-art results on SceneFun3D, largely surpassing both training-free methods and training-based methods, even though without task-specific training and large models.

2 Related Work

Affordance segmentation. Affordance understanding has evolved from object-centric methods [27, 6, 40, 23] focusing on isolated objects, to scene-level reasoning in complex environments. Early 2D affordance methods [39, 21, 27] leverage segmentation models and weakly supervised learning for RGB-based affordance parsing but lack 3D grounding [37] necessary for embodied interaction. Unlike previous 3D affordance datasets focusing on individual objects [12] or closed-vocabulary setting [15], SceneFun3D [11] provides 230 high-resolution real-world indoor scenes with over 3,000 challenging task descriptions requiring world knowledge and spatial reasoning. Fun3DU [7] is the first dedicated method for this benchmark, employing a four-stage training-free pipeline: (1) a text-only LLM (*i.e.*, LLaMA-3.1-9B) performs reasoning to identify contextual and functional objects; (2) open-vocabulary object segmentation locates contextual objects to select views; (3) a VLM grounds functional objects; and (4) geometric lifting aggregates 2D masks into 3D. TASA [16] introduces task-aware frame selection and 3D geometric refinement with learnable components, requiring training on SceneFun3D to optimize for the task. AffordBot [36] operates directly on 3D point clouds rather than videos, rendering surround-view images and fine-tuning Mask3D [31] for 3D instance segmentation. Both require a large MLLM (Qwen2.5-VL-72B). However, these methods face critical limitations: Fun3DU operates initial reasoning without visual input, leading to errors in ambiguous cases. It processes frames independently without leveraging temporal context for spatial disambiguation. Training-based methods like TASA and AffordBot require task-specific data and lack generalization to unseen domains. Specifically, they require point clouds as input and ground-truth point cloud annotations for training.

Instead, UniFunc3D uses a unified MLLM that jointly performs visual reasoning, temporal grounding, and spatial localization in a unified model, eliminating visual blindness and information loss across pipeline stages while remaining fully training-free. Note that there are additional related work sections in the Appendix due to the page limit.

3 Method

3.1 Problem Formulation

Given a 3D scene represented as a point cloud $\mathcal{C} = \{c_i\}_{i=1}^{C}$ where each $c_i \in \mathbb{R}^3$ is a 3D point, and a set of posed RGB-D views $\mathcal{V} = \{v_i\}_{i=1}^{V}$ captured from different viewpoints, along with a natural language task description T, our goal is to segment the functional object F that enables task completion. Unlike explicit queries that directly name objects (*e.g.*, "segment the handle"), 3D affordance segmentation requires inferring: (1) which object to interact with based on world knowledge, (2) which specific part enables the action, and (3) which instance among multiple similar objects satisfies spatial constraints in T. We output a 3D mask $\mathcal{M}_F \subset \mathcal{C}$ indicating which points belong to the functional object.

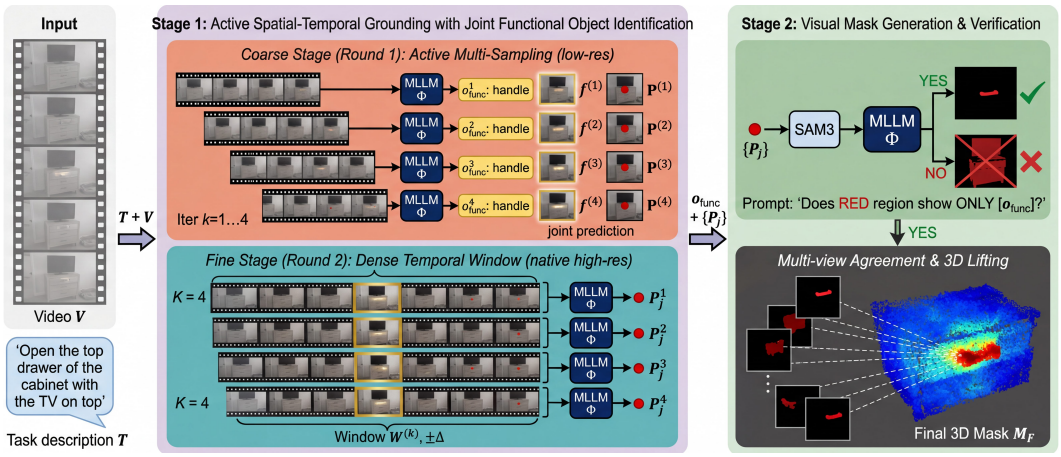


Figure 2: **Method overview.** UniFunc3D employs a unified MLLM with *active spatial-temporal grounding with joint functional object identification*: the coarse stage (Round 1) actively surveys low-resolution video frames across multiple sampling iterations and selects the most informative candidate via visual verification; the fine stage (Round 2) processes a dense temporal window at native high resolution, delivering zoom-in capability while preserving global scene context for precise localization. *Visual mask generation and verification* uses SAM3 for segmentation with MLLM-based mask verification, then multi-view 3D lifting to obtain the final 3D masks.

3.2 Unified MLLM with Visually Grounded Reasoning

Prior work [7] performs task decomposition using a text-only LLM, generating hypotheses about functional objects without visual verification. This leads to errors when objects have ambiguous functional parts that cannot be inferred from language alone. Fig. 2 illustrates the full pipeline. We address this by applying the same MLLM Φ across two visually grounded stages:

$$(o^*_{\text{func}}, \{\mathbf{P}_j\}) = \Phi_{\text{ground}}(T, V), \qquad b^j_i = \Phi_{\text{verify}}(\tilde{I}_j, o^*_{\text{func}})$$

where $\mathcal{V}$ are input video frames, o^*_{func} is the functional object name jointly predicted by the grounding stage, $\mathbf{P}_j$ is the set of affordance points predicted in frame I_j, and $b^j_i \in \{\text{YES}, \text{NO}\}$ is the result of the verification for the mask M^j_i rendered as an overlay image $\tilde{I}_j$. The two stages form a reasoning chain in which each stage consumes visual evidence: **Stage 1 (Active spatial-temporal grounding)** jointly identifies *what* to segment and *where* it is located by simultaneously predicting the functional object name and grounding affordance points across video frames, as detailed in Sec. 3.3; **Stage 2 (Visual mask generation and verification)** generates target masks from the predicted points and visually inspects each segmentation overlay, filtering over-segmented predictions before 3D lifting, as detailed in Sec. 3.4.

3.3 Active Spatial-Temporal Grounding

Existing training-free methods like Fun3DU [7] passively ranks frames by object detection scores for a pre-determined contextual object category, a strategy with no mechanism to adapt when spatial anchors are absent or when fine-grained parts demand higher-resolution inspection.

We address this with **active spatial-temporal grounding**: rather than relying on passive object-detection heuristics, the MLLM acts as an active observer that surveys multiple temporal slices of the video and selects the most informative frames through direct visual evidence, autonomously replacing all hand-crafted rules and hyperparameters. To resolve the small-part visibility challenge without sacrificing global context, we adopt a **human-like coarse-to-fine perception** strategy. A naive crop-and-reprocess zoom-in agent introduces cascaded errors, and an incorrect crop irrecoverably commits the pipeline to a wrong hypothesis. Instead, our coarse stage surveys the full video at low resolution to identify a candidate frame; the fine stage then processes a dense temporal window around it at native high resolution, delivering zoom-in capability while retaining the complete scene so the model can self-correct coarse estimates.

Coarse stage (Round 1): Active multi-sampling frame selection. Given a video sequence $\mathcal{V} = \{v_1, v_2, \ldots, v_L\}$ with L frames, we perform K sampling iterations to ensure robust temporal coverage. For the k-th iteration ($k \in \{1, \ldots, K\}$), we sample N frames at low resolution using uniformly spaced intervals with a shifted starting offset:

$$\mathcal{I}^{(k)} = \{v_{j_1^{(k)}}, v_{j_2^{(k)}}, \ldots, v_{j_N^{(k)}}\}, \quad j_i^{(k)} = \left\lfloor \frac{(i-1) \cdot L}{N} + \frac{k \cdot L}{K \cdot N} \right\rfloor. \tag{1}$$

This offset-based sampling ensures different iterations capture complementary temporal slices of the video. We insert frame index tags `<frame i>:` before each image `i` to enable the MLLM to reference specific frames in its response. Each sampled set $\mathcal{I}^{(k)}$ is fed to the MLLM with a specialized designed prompt (shown in the Appendix Tab. 9), which enables the MLLM to jointly perform semantic reasoning (identifying the functional object), temporal reasoning (selecting which frame best shows it), and spatial grounding (localizing the part), all in a single forward pass. The model outputs a structured response indicating the functional object name, selected frame index, and point coordinates:

$$(o_{\text{func}}^{(k)},\, f^{(k)},\, \mathbf{P}^{(k)}) = \Phi_{\text{R1}}(T,\, \mathcal{I}^{(k)}), \tag{2}$$

where $o_{\text{func}}^{(k)}$ is the predicted functional object name, $f^{(k)}$ is the selected frame index, and $\mathbf{P}^{(k)} = \{(x_i, y_i)\}$ is the set of predicted points. To ensure robust temporal coverage, all K predictions for which a valid frame index is returned are retained and collectively forwarded to the fine stage. This active multi-sampling ensemble prevents any single temporal slice from dominating: diverse offsets collectively cover the full video, and the union of candidate frames from all iterations provides Round 2 with a rich set of starting points for high-resolution refinement.

Fine stage (Round 2): Native zoom-in via dense temporal window at high resolution. The coarse stage produces a set of candidate frames $\mathcal{F} = \{f^{(k)} \mid r_k = \text{valid}\}_{k=1}^{K}$, each from a different temporal offset but all operating at reduced resolution. Mimicking human-like zoom-in perception, for each candidate frame $f^{(k)} \in \mathcal{F}$ the fine stage extracts a dense temporal window $\mathcal{W}^{(k)}$ around it from the full video and processes it at native high resolution:

$$\mathcal{W}^{(k)} = \{v_{f^{(k)}-\Delta}, \ldots, v_{f^{(k)}}, \ldots, v_{f^{(k)}+\Delta}\}, \tag{3}$$

where Δ defines the window radius. The temporal window is calculated based on the sampling interval: if Round 1 samples N frames from a video of duration t_{total}, each interval spans $\Delta t = t_{\text{total}}/(N-1)$, and we extract frames within $\pm\Delta t/2$ seconds of the candidate frame's timestamp. For each frame $I_j \in \mathcal{W}^{(k)}$, we query the MLLM independently with a high resolution image (as detailed in the Appendix Tab. 10): This enables fine-grained localization for each frame independently. Processing all frames in $\mathcal{W}^{(k)}$ provides multiple candidate predictions per window:

$$\{\mathbf{P}_j\}_{j \in \mathcal{W}^{(k)}} = \{\Phi_{\text{R2}}(T,\, o_{\text{func}}^{j},\, I_j)\}_{j \in \mathcal{W}^{(k)}}, \tag{4}$$

where o_{func}^{j} is the functional object predicted by the coarse stage. This is repeated for all K candidate windows, yielding a combined set of per-frame point predictions across all temporal windows.

The fine stage addresses all three failure modes of passive frame selection by providing dense multi-view context from multiple angles and timestamps at high resolution: (1) Missing spatial context becomes visible across the temporal window as multiple viewpoints collectively cover the full scene context. (2) Partially visible targets gain complete coverage through multi-view aggregation, resolving spatial ambiguities that are unresolvable in any single isolated frame. (3) The resolution transition from the coarse stage to native high resolution delivers zoom-in capability for small functional parts without any image cropping: because the full frame is retained rather than cropped, the surrounding context remains available for disambiguation throughout. Crucially, if the coarse stage's initial estimate is slightly off-target, the fine stage can self-correct by reasoning over the complete high-resolution scene, a recovery that is impossible in a crop-and-reprocess agent once the wrong region has been committed to. We aggregate per-frame predictions from all K temporal windows into a combined set $\{\mathbf{P}_j\}$ that leverages holistic spatial-temporal understanding.

3.4 Visual Mask Generation and Verification

From the fine stage, we obtain per-frame point predictions $\{\mathbf{P}_j\}$ aggregated across all K temporal windows, where each $\mathbf{P}_j = \{(x_i^j, y_i^j)\}_{i=1}^{M_j}$ contains M_j points. We use them as point prompts for

SAM3 [2]. For each point prompt $p_i^j = (x_i^j, y_i^j)$ in frame I_j across all K windows, it generates a binary mask M_i^j. As SAM3 may occasionally over-segment by including parent objects, we apply MLLM-based visual mask verification as the final stage of the unified reasoning framework. Rather than verifying points through text-level queries (as in TASA [16]), we render each mask as a colored overlay on its source frame and present it directly to the MLLM for visual judgment, enabling the model to reason about whether the highlighted region is precisely the intended functional part. We create an overlay image $\tilde{I}_j = \text{Overlay}(I_j, M_i^j, \alpha)$ by highlighting the masked region with transparency α. The verification is formulated as:

$$b_i^j = \Phi_{\text{verify}}(\tilde{I}_j, o^*_{\text{func}}) \in \{\text{YES}, \text{NO}\}. \tag{5}$$

The model verifies two criteria: (1) The highlighted region is the target functional object. (2) It does not include parent or containing objects. Detailed process is described in the Appendix. Only masks receiving $b_i^j = \text{YES}$ are retained for 3D lifting $\mathcal{M}_{\text{verified}}$. By reusing the same MLLM backbone that performed Stage 1, the verification maintains semantic consistency throughout the unified reasoning pipeline and substantially improves precision without sacrificing recall.

3.5 Multi-view Agreement and 3D Lifting

To obtain the final 3D masks, following [7], we lift the verified 2D masks from all processed frames onto the point cloud using camera poses. We employ multi-view agreement to filter spurious predictions: For each 3D point $c_i \in \mathcal{C}$, we count how many 2D masks project onto it:

$$s_i = \sum_{k=1}^{K} |\{p^k \mid \Gamma^k(p^k) = c_i, p^k \in M^k\}|, \tag{6}$$

where M^k is the 2D functional mask in the k-th view, p^k is a pixel in M^k, and $\Gamma^k \colon \mathbb{Z}^2 \to \mathbb{R}^3$ maps 2D pixels to 3D points via depth and pose. We normalize $s_i \in [0, 1]$ and threshold at τ to produce the final mask $\mathcal{M}_F = \{c_i \mid s_i > \tau\}$. This consensus mechanism suppresses outliers and ensures that only points consistently identified across multiple views are included.

4 Experiments

4.1 Experimental Setup

Dataset. We evaluate on SceneFun3D [11], the only dataset for affordance segmentation in 3D scenes. It contains 230 high-resolution indoor scene scans split into *split0* (30 scenes, from their validation set) and *split1* (200 scenes, from their training set), following the practice of Fun3DU.

Evaluation metrics. Following the original SceneFun3D [11] benchmark design, we report Average Precision at IoU thresholds of 0.25 and 0.5 (AP_{25}, AP_{50}). For comprehensive evaluation, we additionally report Average Recall (AR_{25}, AR_{50}) and mean Intersection-over-Union (mIoU). These metrics capture both localization accuracy and segmentation quality.

Implementation details. We implement two variants (8B, 30B) of UniFunc3D using MLLM models in different sizes of Qwen3-VL variants [1, 9]. For Round 1, we sample $N = 64$ frames per video at 512×384 resolution with $K = 4$ sampling iterations using different temporal offsets. For Round 2, we extract temporal windows of size $2\Delta + 1$ frames around the selected frame. We use SAM3 [2] for mask generation with a confidence threshold of 0.5. The overlay transparent parameter α is set to 0.7. The final 3D mask is obtained by thresholding the multi-view agreement heatmap at $\tau = 0.7$. All models are frozen and used without fine-tuning, making UniFunc3D fully training-free.

Baselines. We compare against both training-free and training-based methods: (1) OV-3D segmentation methods **OpenMask3D** [33], **LERF** [18], and **OpenIns3D** [17], which lack task-specific reasoning. (2) Training-based methods **TASA** [16] and **AffordBot** [36], which utilize Qwen2.5-VL-72B and require training on 3D labels. (3) **Fun3DU** [7], the pioneering training-free method.

4.2 Main Results

Tab. 1 presents quantitative results on SceneFun3D. Our UniFunc3D-30B achieves the best performance across **all** methods on **all** reported metrics on **both** splits. Remarkably, our training-free approach even outperforms training-based methods, including those using significantly larger models.

Table 1: **Results on SceneFun3D.** UniFunc3D achieves the best performance across all metrics on both split0 and split1, outperforming both training-free and training-based methods. Percentages in parentheses denote relative improvements over Fun3DU.

	Split0 (30 scenes, val)					Split1 (200 scenes, train)				
Method	AP_{50} ↑	AP_{25} ↑	AR_{50} ↑	AR_{25} ↑	mIoU↑	AP_{50} ↑	AP_{25} ↑	AR_{50} ↑	AR_{25} ↑	mIoU↑
Open-vocabulary 3D segmentation methods:										
OpenMask3D [33]	0.2	0.4	24.5	27.0	0.2	0.0	0.0	1.4	2.6	0.1
LERF [18]	0.0	0.0	35.1	36.0	0.0	0.0	0.0	24.6	25.1	0.0
OpenIns3D [17]	0.0	0.0	46.7	51.5	0.1	0.0	0.0	37.1	39.9	0.1
Training-based 3D affordance segmentation methods:										
TASA-72B [16]	26.9	28.6	–	–	19.7	*trained on split1; not evaluated*				
AffordBot-72B [36]	20.91	24.76	18.99	22.84	14.42	*trained on split1; not evaluated*				
Training-free 3D affordance segmentation methods:										
Fun3DU† [7]	16.9	33.3	38.2	46.7	15.2	12.6	23.1	32.9	40.5	11.5
UniFunc3D-8B	23.82 (+40.9%)	44.04 (+32.3%)	46.07 (+20.6%)	55.51 (+18.9%)	20.92 (+37.6%)	16.24 (+28.9%)	29.02 (+25.6%)	38.91 (+18.3%)	48.15 (+18.9%)	14.23 (+23.7%)
UniFunc3D-30B	**31.24** (+84.9%)	**51.01** (+53.2%)	**46.97** (+23.0%)	**58.88** (+26.1%)	**24.30** (+59.9%)	**21.32** (+69.2%)	**35.76** (+54.8%)	**40.03** (+21.7%)	**51.00** (+25.9%)	**17.09** (+48.6%)

† Fun3DU adopts a sequential offline multi-model pipeline that includes a text-only LLM inference with LLaMa-3.1-9B [14], contextual object detection with OWLv2 [24] and RobustSAM [4], Molmo-7B [10] grounding the functional objects on top-ranked views, and SAM [19]. Thus, it may be improper to list its "total" number of parameters like others in this table.

Comparison with training-free methods. Compared to Fun3DU, the previous state-of-the-art training-free method, UniFunc3D-30B improves AP_{50} by 14.34 points (84.9% relative gain) and AP_{25} by 17.71 points (53.2% gain) on split0, showing significantly superior localization accuracy at both tight and loose IoU thresholds. The mIoU improvement of 9.1 points (59.9% gain over Fun3DU) further confirms more accurate segmentation of functional parts. UniFunc3D-8B also demonstrates strong performance, improving AP_{50} by 6.92 points (41.0% gain) and AP_{25} by 10.74 points (32.3% gain) over Fun3DU on split0.

Comparison with training-based methods. Most notably, UniFunc3D-30B substantially outperforms training-based methods despite being entirely training-free and with a smaller model (30B vs. 72B). Against AffordBot-72B, which requires 1,000 epochs of fine-tuning on SceneFun3D training data, UniFunc3D-30B demonstrates even more striking advantages: AP_{50} (+10.33 points, 49.4% gain), AP_{25} (+26.25 points, 106.0% gain), and mIoU (+9.88 points, 68.5% gain). These results demonstrate that our unified, visually aware multimodal architecture with temporal grounding is fundamentally more effective than existing specialized training-based approaches.

Open-vocabulary methods. Open-vocabulary methods achieve moderate recall but near-zero precision, indicating they over-segment. They lack the reasoning capability to interpret implicit task descriptions, demonstrating the necessity of task-aware reasoning for 3D affordance segmentation.

Qualitative results. Fig. 3 compares AffordBot, Fun3DU, and our method against ground truth on five representative queries. Our method clearly outperforms others in solving spatial disambiguation and handling small objects with diverse affordance. For example, for input prompt `Open the top left drawer of the cabinet with the beauty products on top`, our method can find the correct `top-left knob`, while AffordBot finds the wrong `top-right knob` and Fun3DU mistakenly treats the `drawer` as the functional object.

4.3 Ablation Study

We conduct ablation studies on SceneFun3D split0 using UniFunc3D-8B.

Vanilla MLLMs vs. two-stage grounding. Tab. 2 shows that the vanilla MLLM (*i.e.*, one-stage, directly using Qwen3-VL-8B without other designs except the external segmentation model) approach processing all frames at high resolution without initial coarse selection, severely degrades performance despite using the highest resolution throughout. This highlights that **directly using off-the-shelf MLLMs CANNOT solve this problem**: when all frames are processed at high resolution, the model must handle fine-grained detail while lacking the global temporal overview needed to identify the



Figure 3: **Qualitative comparison.** We show results for representive five queries.

Table 2: **Ablation on high-level designs.** Comparison of one-stage vs. two-stage approaches. We also evaluate the impact of **Multi-Sam.** (Multi-sampling), and **Temp.** (Temporal window).

Strategy	2-Stage	Multi-Sam.	Temp.	AP_{50}	AP_{25}	AR_{50}	AR_{25}	mIoU
Qwen3-VL-8B (High-res)				4.27	7.64	24.27	28.76	3.11
Qwen3-VL-8B (Low-res)				7.42	14.38	33.93	38.65	7.35
Low → High (Zoom-in)	✓			15.96	26.29	39.33	48.54	12.71
Two-stage + Multi-sam.	✓	✓		19.33	38.20	44.94	53.71	18.10
Two-stage + Temp. window	✓		✓	17.53	35.28	41.80	51.91	17.26
Ours (Full)	✓	✓	✓	**23.82**	**44.04**	**46.07**	**55.51**	**20.92**

correct frame, and base MLLMs (*i.e.*, Qwen) degrade under such extremely long contexts. One-stage low-resolution processing is equally limited; although the shorter context better utilizes model capacity, individual frames lack the spatial detail required for precise functional part localization. This matches the practical constraints of using MLLMs.

Our proposed two-stage approach with a single Round 2 frame (low→high) substantially improves performance, confirming the value of coarse-to-fine processing. Adding multisampling alone improves further by diversifying Round 1 coverage across shifted temporal offsets and reducing the risk of missing the optimal frame. Adding temporal window processing alone also yields gains by expanding Round 2 context from a single frame to multiple frames. Our full model combines both components, achieving the best performance, validating that both components provide complementary benefits.

Table 3: **Ablation on sampling strategies.** Results for UniFunc3D-8B evaluating the impact of sampling iterations (K), the temporal window and verification.

Configuration	Temp.	Verif.	AP_{50}	AP_{25}	AR_{50}	AR_{25}	mIoU
$K = 1$ (Base)			15.96	26.29	39.33	48.54	12.71
$K = 1$, no verify	✓		15.06	32.58	40.90	50.11	16.00
$K = 1$, verify	✓	✓	17.53	35.28	41.80	51.91	17.26
$K = 2$,			15.51	28.31	39.33	46.07	14.40
$K = 2$, no window		✓	12.58	27.42	40.00	47.42	13.87
$K = 2$, no verify	✓		21.12	39.10	**46.97**	56.18	19.57
$K = 2$, verify	✓	✓	22.02	43.60	46.29	54.38	20.69
$K = 4$,			15.96	30.11	41.57	50.11	15.30
$K = 4$, no window		✓	19.33	38.20	44.94	53.71	18.10
$K = 4$, no verify	✓		21.35	41.12	46.74	55.51	19.99
$K = 4$, verify	✓	✓	**23.82**	**44.04**	46.07	55.51	**20.92**
$K = 8$, no window		✓	22.47	42.70	42.70	52.81	19.48
$K = 8$, no verify	✓		21.57	42.25	46.52	**56.40**	19.95



(a) Ablation ($K = 1$) (b) UniFunc3D (Ours) (c) Ground Truth

Figure 4: Visual comparison. Input prompt: `open the right door of the wooden display cabinet to the left of the paintings`.

Effect of temporal window. The temporal window provides the largest single gain. Adding the temporal window when $K = 2$ raises AP_{50} to 21.12 (+5.61) and AP_{25} from 28.31 to 39.10 (+10.79). The same trend holds at $K = 4$. The temporal window also critically enables verification to function effectively: applying verification without the window at $K = 2$ slightly *hurts* performance, because there are too few candidate frames for reliable confidence ranking.

Effect of sampling iterations (K). With the temporal window enabled, increasing from $K = 1$ to $K = 2$ provides substantial gains, as multiple shifted offsets reduce the risk of missing the optimal frame region. Further increasing to $K = 4$ yields only marginal improvement over $K = 2$ without verification (21.35 vs. 21.12 AP_{50}), and $K = 8$ shows similarly diminishing returns (21.57 AP_{50}), suggesting $K = 4$ is the practical saturation point. Fig. 4 shows that single-iteration sampling ($K = 1$) fails due to limited views. Without diverse temporal candidates, the model may produce spatially inconsistent predictions, such as misidentifying an adjacent handle. Our multi-sampling strategy resolves these local ambiguities by providing broader visual evidence.

Effect of verification. Verification is most effective when the candidate pool is large enough. With $K = 2$, it slightly improves AP_{50} (22.02 vs. 21.12) and substantially improves AP_{25} (43.60 vs. 39.10). With $K = 4$, the gains are more consistent: +2.47 AP_{50}, +2.92 AP_{25}, and +0.93 mIoU, as the larger pool gives the filter more room to select high-confidence frames.

Effect of base models. To evaluate the robustness of our approach, we conduct an ablation experiment on the base MLLM models we used. For a fair comparison, we adopt Qwen2.5-VL-72B [34] as our backbone, aligning with the architecture used by contemporary training-based baselines, specifically TASA [16] and AffordBot [36]. As shown in Tab. 4, the experimental results demonstrate that our training-free model, when integrated with Qwen2.5-VL-72B, consistently outperforms existing training-based methods across nearly all evaluation protocols.

Table 4: **Ablation on base models.** Best-performed metrics with Qwen2.5-VL-72B (the top group) are marked in **bold**. We also provide our models with Qwen3-VL for reference.

Baselines	AP_{50}	AP_{25}	AR_{50}	AR_{25}	mIoU
TASA-72B† [16]	**26.9**	28.6	–	–	19.7
AffordBot-72B† [36]	20.91	24.76	18.99	22.84	14.42
Ours (Qwen2.5-VL-72B)	22.02	**40.90**	**48.09**	**59.78**	**20.07**
Ours (Qwen3VL-8B)	23.82	44.04	46.07	55.51	20.92
Ours (Qwen3VL-30B)	31.24	51.01	46.97	58.88	24.30

† Qwen2.5-VL-72B is used according to the authors.

Table 5: **Ablation on verification models.**

Baselines	AP_{50}	AP_{25}	AR_{50}	AR_{25}	mIoU
SAM [19]	23.56	**44.23**	45.91	54.99	20.78
SAM3 [2]	**23.82**	44.04	**46.07**	**55.51**	**20.92**

Table 6: **Ablation on radius settings.**

Radius	AP_{50}	AP_{25}	AR_{50}	AR_{25}	mIoU
0.5Δ	22.92	**44.94**	45.84	55.28	20.80
1.0Δ	23.82	44.04	**46.07**	55.51	20.92
1.5Δ	**24.49**	44.27	45.62	**55.96**	**20.94**

Specifically, we surpass TASA in 2 out of 3 metrics, including AP_{50}, AP_{25}, and mIoU, and exceed AffordBot's performance across AP_{50}, AP_{25}, AR_{50}, AR_{25}, and mIoU. While TASA maintains a marginal lead in AP_{50} in one specific configuration, our model's superiority across diverse metrics underscores the efficacy of our proposed pipeline enhancements. This shows the generalizability of our framework, proving that its performance gains are not merely a byproduct of a superior base model but stem from our intrinsic methodological design.

Effect of verification models. To verify the design of our verification models, we change the segmentation model we used (*i.e.*, SAM3) to the previous SAM [19], used by Fun3DU [7]. As shown in Tab. 5, We do not observe significant improvement by changing the segmentation model. This indicates that the most improvement of our UniFunc3D does **NOT** come from the modern or powerful segmentor but the overall pipeline design.

Effect of hyperparameter Δ. Table 6 shows the effect of hyperparameter Δ, which is relatively insensitive within this range, whereas the number of shifted sampling iterations K has a larger effect.

5 Conclusion

We presented UniFunc3D, a unified framework for 3D scene affordance segmentation that consolidates semantic reasoning, temporal grounding, and spatial localization into a single multimodal large language model. By processing video frames with our active spatial-temporal grounding process, UniFunc3D eliminates the visual blindness and cascading errors inherent in fragmented multi-model pipelines. Extensive experiments show that UniFunc3D achieves state-of-the-art performance across all metrics, surpassing both training-free and training-based methods by a large margin.

Acknowledgments and Disclosure of Funding

The research is supported in part by the Early Career Scheme of the Research Grants Council (RGC) of the Hong Kong SAR under grant No. 26202321, ITF PRP/046/24FX, Science & Technology Cooperation Program of Shandong under grant No. SDST26EG01, SAIL Research Project, and HKUST-Zeekr Collaborative Research Fund. We sincerely thank Shuquan Ye for the discussions and for her generous personal hardware support for this project.

References

[1] S. Bai, Y. Cai, R. Chen, K. Chen, X. Chen, Z. Cheng, L. Deng, W. Ding, C. Gao, C. Ge, W. Ge, Z. Guo, Q. Huang, J. Huang, F. Huang, B. Hui, S. Jiang, Z. Li, M. Li, M. Li, K. Li, Z. Lin, J. Lin, X. Liu, J. Liu, C. Liu, Y. Liu, D. Liu, S. Liu, D. Lu, R. Luo, C. Lv, R. Men, L. Meng, X. Ren, X. Ren, S. Song, Y. Sun, J. Tang, J. Tu, J. Wan, P. Wang, P. Wang, Q. Wang, Y. Wang, T. Xie, Y. Xu, H. Xu, J. Xu, Z. Yang, M. Yang, J. Yang, A. Yang, B. Yu, F. Zhang, H. Zhang, X. Zhang, B. Zheng, H. Zhong, J. Zhou, F. Zhou, J. Zhou, Y. Zhu, and K. Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[2] N. Carion, L. Gustafson, Y.-T. Hu, S. Debnath, R. Hu, D. Suris, C. Ryali, K. V. Alwala, H. Khedr, A. Huang, J. Lei, T. Ma, B. Guo, A. Kalla, M. Marks, J. Greer, M. Wang, P. Sun, R. Rädle, T. Afouras, E. Mavroudi, K. Xu, T.-H. Wu, Y. Zhou, L. Momeni, R. Hazra, S. Ding, S. Vaze, F. Porcher, F. Li, S. Li, A. Kamath, H. K. Cheng, P. Dollár, N. Ravi, K. Saenko, P. Zhang, and C. Feichtenhofer. Sam 3: Segment anything with concepts, 2025.

[3] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niebner, M. Savva, S. Song, A. Zeng, and Y. Zhang. Matterport3D: Learning from RGB-D Data in Indoor Environments. In *3DV*, 2017.

[4] W.-T. Chen, Y.-J. Vong, S.-Y. Kuo, S. Ma, and J. Wang. Robustsam: Segment anything robustly on degraded images. 2024.

[5] C. Choy, J. Gwak, and S. Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3075–3084, 2019.

[6] H. Chu, X. Deng, Q. Lv, X. Chen, Y. Li, J. Hao, and L. Nie. 3d-affordancellm: Harnessing large language models for open-vocabulary affordance detection in 3d worlds. *arXiv preprint arXiv:2502.20041*, 2025.

[7] J. Corsetti, F. Giuliari, A. Fasoli, D. Boscaini, and F. Poiesi. Functionality understanding and segmentation in 3d scenes. In *CVPR*, pages 24550–24559, 2025.

[8] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017.

[9] R. Dang, J. Guo, B. Hou, S. Leng, K. Li, X. Li, J. Liu, Y. Mao, Z. Wang, Y. Yuan, M. Zhu, X. Lin, Y. Bai, Q. Jiang, Y. Zhao, M. Zeng, J. Gao, Y. Jiang, J. Cen, S. Huang, L. Wang, W. Zhang, C. Liu, J. Yang, S. Lu, and D. Zhao. Rynnbrain: Open embodied foundation models, 2026.

[10] M. Deitke, C. Clark, S. Lee, R. Tripathi, Y. Yang, J. S. Park, M. Salehi, N. Muennighoff, K. Lo, L. Soldaini, J. Lu, T. Anderson, E. Bransom, K. Ehsani, H. Ngo, Y. Chen, A. Patel, M. Yatskar, C. Callison-Burch, A. Head, R. Hendrix, F. Bastani, E. VanderBilt, N. Lambert, Y. Chou, A. Chheda, J. Sparks, S. Skjonsberg, M. Schmitz, A. Sarnat, B. Bischoff, P. Walsh, C. Newell, P. Wolters, T. Gupta, K.-H. Zeng, J. Borchardt, D. Groeneveld, C. Nam, S. Lebrecht, C. Wittlif, C. Schoenick, O. Michel, R. Krishna, L. Weihs, N. A. Smith, H. Hajishirzi, R. Girshick, A. Farhadi, and A. Kembhavi. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 91–104, June 2025.

[11] A. Delitzas, A. Takmaz, F. Tombari, R. Sumner, M. Pollefeys, and F. Engelmann. SceneFun3D: Fine-grained functionality and affordance understanding in 3D scenes. In *CVPR*, 2024.

[12] S. Deng, X. Xu, C. Wu, K. Chen, and K. Jia. 3d affordancenet: A benchmark for visual object affordance understanding. In *CVPR*, 2021.

[13] F. Engelmann, F. Manhardt, M. Niemeyer, K. Tateno, M. Pollefeys, and F. Tombari. Opennerf: Open set 3d neural scene segmentation with pixel-wise features and rendered novel views. *arXiv preprint arXiv:2404.03650*, 2024.

[14] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[15] A.-M. Halacheva, Y. Miao, J.-N. Zaech, X. Wang, L. Van Gool, and D. P. Paudel. Articulate3d: Holistic understanding of 3d scenes as universal scene description. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5633–5644, 2025.

[16] L. He, M. Liu, Q. Ye, Y. Zhou, X. Deng, and G. Ding. Task-aware 3d affordance segmentation via 2d guidance and geometric refinement. *AAAI*, 2026. To appear.

[17] Z. Huang, X. Wu, X. Chen, H. Zhao, L. Zhu, and J. Lasenby. Openins3d: Snap and lookup for 3d open-vocabulary instance segmentation. In *European Conference on Computer Vision*, pages 169–185. Springer, 2024.

[18] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 19729–19739, 2023.

[19] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[20] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

[21] G. Li, D. Sun, L. Sevilla-Lara, and V. Jampani. One-shot open affordance learning with foundation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3086–3096, 2024.

[22] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[23] D. Lu, L. Kong, T. Huang, and G. H. Lee. Geal: Generalizable 3d affordance learning with cross-modal consistency. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1680–1690, 2025.

[24] M. Minderer, A. Gritsenko, and N. Houlsby. Scaling open-vocabulary object detection. *Advances in Neural Information Processing Systems*, 36:72983–73007, 2023.

[25] P. Nguyen, T. D. Ngo, E. Kalogerakis, C. Gan, A. Tran, C. Pham, and K. Nguyen. Open3dis: Open-vocabulary 3d instance segmentation with 2d mask guidance. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4018–4028, 2024.

[26] S. Peng, K. Genova, C. Jiang, A. Tagliasacchi, M. Pollefeys, T. Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 815–824, 2023.

[27] S. Qian, W. Chen, M. Bai, X. Zhou, Z. Tu, and L. E. Li. Affordancellm: Grounding affordance from vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7587–7597, 2024.

[28] M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister. Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20051–20060, 2024.

[29] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[30] D. Rotondi, F. Scaparro, H. Blum, and K. O. Arras. Fungraph: Functionality aware 3d scene graphs for language-prompted scene interaction. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

[31] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe. Mask3d: Mask transformer for 3d semantic instance segmentation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8216–8223. IEEE, 2023.

[32] H. Tai, Q. He, J. Zhang, Y. Qian, Z. Zhang, X. Hu, X. Li, Y. Wang, and Y. Liu. Open-vocabulary sam3d: Towards training-free open-vocabulary 3d scene understanding. *arXiv preprint arXiv:2405.15580*, 2024.

[33] A. Takmaz, E. Fedele, R. W. Sumner, M. Pollefeys, F. Tombari, and F. Engelmann. Openmask3d: Open-vocabulary 3d instance segmentation. *arXiv preprint arXiv:2306.13631*, 2023.

[34] Q. Team. Qwen2.5-vl, January 2025.

[35] T. Ton, J. W. Hong, S. Eom, J. Y. Shim, J. Kim, and C. D. Yoo. Zero-shot dual-path integration framework for open-vocabulary 3d instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7598–7607, 2024.

[36] X. Wang, X. Yang, Y. Xu, Y. Wu, Z. Li, and N. Zhao. Affordbot: 3d fine-grained embodied reasoning via multimodal large language models. *NeurIPS*, 2025.

[37] Y. Wang, L. Ke, B. Zhang, T. Qu, H. Yu, Z. Huang, M. Yu, D. Xu, and D. Yu. N3d-vlm: Native 3d grounding enables accurate spatial reasoning in vision-language models. *arXiv preprint arXiv:2512.16561*, 2025.

[38] A. Werby, D. Rotondi, F. Scaparro, and K. O. Arras. Keysg: Hierarchical keyframe-based 3d scene graphs. *arXiv preprint arXiv:2510.01049*, 2025.

[39] L. Xu, Y. Gao, W. Song, and A. Hao. Weakly supervised multimodal affordance grounding for egocentric images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 6324–6332, 2024.

[40] Y. Yang, W. Zhai, H. Luo, Y. Cao, J. Luo, and Z.-J. Zha. Grounding 3d object affordance from 2d interactions in images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10905–10915, 2023.

[41] Y. Yin, Y. Liu, Y. Xiao, D. Cohen-Or, J. Huang, and B. Chen. Sai3d: Segment any instance in 3d scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3292–3302, 2024.

[42] C. Zhang, A. Delitzas, F. Wang, R. Zhang, X. Ji, M. Pollefeys, and F. Engelmann. Open-vocabulary functional 3d scene graphs for real-world indoor spaces. In *CVPR*, 2025.

[43] X. Zou, J. Yang, H. Zhang, F. Li, L. Li, J. Wang, L. Wang, J. Gao, and Y. J. Lee. Segment everything everywhere all at once. *Advances in neural information processing systems*, 36:19769–19782, 2023.

A Additional Experiments and Discussions

A.1 Efficiency Analysis

We conduct an efficiency analysis with training-free methods Fun3DU [7] and our UniFunc3D. We omit metrics such as total parameters or FLOPs, as the multi-model design of Fun3DU makes a unified estimation of these values inherently imprecise. Furthermore, the standard FLOP metric fails to account for critical non-neural operations essential to the Fun3DU pipeline, such as 3D lifting and point cloud aggregation. These geometric transformations contribute significantly to the overall execution latency but are not reflected in traditional neural network complexity measures, making end-to-end inference time a more representative benchmark for efficiency. Thus, we adopt Total Inference Latency (*i.e.*, time per scene and time per query), a simple but better metric, for comparing such heterogeneous architectures based on the same platform. Besides, we exclude training-based methods (e.g., TASA [16] and AffordBot [36]) from this specific analysis. This is due to the partial unavailability of their source code, specifically the training code for AffordBot's 3D refinement, making a reproducible comparison impossible in this regard.

Table 7: **Time comparison.** UniFunc3D-8B achieves over 5.2 times speedup compared to Fun3DU while maintaining superior accuracy.

Method	Time per scene	Time per query	Speedup
Fun3DU [7]	∼82 min	∼332 sec	1×
UniFunc3D-8B	∼16 min	∼53 sec	**5.2**×
UniFunc3D-30B	∼26 min	∼99 sec	**3.2**×

Tab. 7 compares the total processing time averaged over the validation split (30 scenes) on the same platform with eight H100s. Fun3DU requires approximately 82 minutes per scene due to its multi-stage pipeline using different models: text-only LLM inference with LLaMa-3.1-9B [14], contextual object detection with OWLv2 [24] and RobustSAM [4] on hundreds of input video frames, Molmo-7B [10] grounding the functional objects on top-ranked views, and then do functional object segmentation by SAM [19]. UniFunc3D reduces this to just 26 minutes per scene, a ∼3.2 times speedup, by consolidating reasoning and perception into a single MLLM and using the two-round temporal strategy to avoid exhaustive frame processing. Another benefit from our single MLLM design is that it can better utilize the advantages of some modern efficient inference and serving engines for LLMs such as VLLM [20] with PagedAttention and KV-cache. This efficiency gain is achieved while simultaneously improving accuracy, demonstrating the advantages of our unified architecture.

A.2 Experimental Details

All experiments are conducted on a high-performance computing node equipped with an Intel Xeon Platinum 8468 (2.1GHz, 48-core) CPU and eight NVIDIA H100 GPUs. To ensure a rigorous and controlled environment, all baselines and our proposed model are evaluated on this same hardware platform. It is worth noting that our method can still run on a single H100 but we do not have enough time to re-run all baseline and ablation experiments on a different platform, thus leaving the experiments to the community when the code is available in the near future. For the base model we use in UniFunc3D, we adopt Qwen2.5-VL [34] and a series variant [9] of Qwen3-VL with the same amount of parameter. Note that we do not observe significant improvement from the variants when compared to the vanilla Qwen model [1].

For the baseline results presented in the main paper, we follow a best-of-both strategy: we report the higher performance between the values provided in the original papers and our own reproduced results if their source code is available. For instance, as shown in Tab. 8, our reproduced results of AffordBot [36] achieved slightly higher performance than the results reported in the original work. For TASA [16] and Fun3DU [7], the reported results are better than our reproduced results. Consequently, we adopt these improved metrics for our main comparisons to ensure that our performance gains are measured against the strongest possible version of the state-of-the-art. This conservative reporting approach ensures that the observed improvements of UniFunc3D are not attributed to suboptimal baseline configurations but rather to our architectural innovations.

Table 8: **Comparative results of different implementations of baseline methods.**

Implementation	AP_{50}	AP_{25}	AR_{50}	AR_{25}	mIoU
TASA-72B (Reported)	**26.9**	**28.6**	–	–	**19.7**
TASA-72B (Our Reproduced)	19.72	23.58	21.17	25.21	14.61
AffordBot-72B (Reported) [36]	20.00	23.30	-	-	14.00
AffordBot-72B (Our Reproduced)	**20.91**	**24.76**	**18.99**	**22.84**	**14.42**
Fun3DU (Reported) [7]	**16.9**	**33.3**	**38.2**	**46.7**	**15.2**
Fun3DU (Our Reproduced)	12.13	23.15	33.93	40.90	11.34

Table 9 and Table 10 show the system prompt used in the coarse stage and the fine stage of our proposed method, respectively.

```
Given these frames and the task: [T], complete three tasks:
1. Identify the functional object needed to accomplish the task.
2. Select the key frame that best shows the functional object.
3. Identify a single affordance point (x, y) on the functional
object.
Output format: <affordance> functional object: ...; <frame n>: ...;
(x, y) </affordance>
```

Table 9: Prompt used in the coarse grounding stage of our UniFunc3D. T is the input task prompt.

```
"Identify the affordance point on the [o*_func] in order to [T].
Output format: <affordance> (x, y) </affordance>."
```

Table 10: Prompt used in the fine grounding stage of our UniFunc3D. T is the input task prompt. o^j_{func} is the functional object predicted by the coarse stage.

A.3 Limitations and Future Work

While UniFunc3D significantly advances the state-of-the-art, challenges remain for extremely small functional parts ($<$0.1% of image area) or scenes with severe occlusion. Future work could explore different zoom-in mechanisms or incorporate explicit 3D geometric reasoning directly within the MLLM to handle complex spatial layouts. Additionally, extending UniFunc3D to support interactive refinement in which users provide feedback to correct predictions would be valuable for real-world robotic applications.

A.4 Additional Visual Comparisons for Ablation Models

Fig. 5 shows a further visual comparison for ablation models, which removes the temporal window, multi-sampling and verification, respectively. For input prompt: `Open the top left drawer of the cabinet with the TV on top`, the base model wrongly predicts expansive and inaccurate masks of the top left drawer of the left cabinet, but not the one with the TV on top, resulting in an mIoU of 0.000, as it fails to isolate the fine-grained functional part of the knob due to the limited size of selected frame candidates for correct spatial-temporal reasoning. Similarly, removing the temporal window or multi-sampling leads to mislocalization or missing detections entirely, though it can nearly locate a knob, but not the correct one. However, after removing verification, the model will mistakenly treat the drawer as the target functional object instead, due to the absence of verification to check if the mask is the exact functional object type `knob`. In contrast, our full UniFunc3D model achieves an mIoU of 0.523 by correctly identifying the precise target region, aligning closely with the ground truth. This visual evidence confirms that our unified reasoning-perception approach and temporal strategies are essential for robust 3D affordance grounding, particularly for fine-grained parts where simpler baselines often suffer from over-segmentation or localization failure.

`Does the RED highlighted region show ONLY [`o^*_{func}`]?`

Table 11: Prompt used in the verification stage of our UniFunc3D. o^*_{func} is the functional object predicted by the previous stage.

Similarly, Fig. 6 shows another further visual comparison for ablation models, given an input text `Open the right door of the wood and glass display cabinet next to the pet transport box`. We can see that without multi-sampling and temporal window the model cannot locate the correct `knob` (both mIoU: 0.00) due to the limited number of selected frame candidates with potential targets and lack good spatial-temporal reasoning during this process. While the ablation model without verification can find the correct knob, it will treat some other points as the target object (at the bottom corner of the cabinet), resulting in an mIoU of 0.058. Our method can achieve the best results (mIoU: 0.476) with the help of sufficient spatial-temporal reasoning and the final verification. These examples further demonstrate the effectiveness of our proposed pipelines.

B Additional Related Work

Open-vocabulary 3D segmentation. Open-vocabulary 3D (OV-3D) segmentation methods [26, 33, 41, 35, 25, 17] aim to segment objects in 3D scenes using natural language descriptions. These approaches typically combine 3D proposal modules for predicting masks from point clouds [5, 31] with 2D modules that extract masks from multi-view RGB images using VLMs [29, 22] and segmentation models [19, 43]. Segmentation is achieved through 2D-3D fusion based on mask agreement [33, 32, 35] or learnable pooling [25, 26]. Language-guided radiance field methods [18, 28, 13] can also perform OV3DS but require scene-specific training. Empirical results on SceneFun3D [11] demonstrate that these methods struggle with affordance segmentation, as they rely on modules pre-trained on 3D datasets [3, 8] biased toward large furniture rather than small functional parts, and they interpret concise descriptions with explicit object names rather than implicit, context-dependent task descriptions. Moreover, these methods lack mechanisms to disambiguate multiple instances of the same object class (*e.g.*, selecting the correct cabinet when multiple cabinets exist) or to leverage multi-frame temporal context for spatial referring expressions. UniFunc3D addresses these limitations by using an MLLM to interpret complex task descriptions with visual evidence and performing video-based temporal grounding to resolve spatial ambiguities across frames.

Applications on 3D functionality scene graph. FunGraph [30] augments 3D scene graphs with functional elements for language-prompted interaction. OpenFunGraph [42] predicts open-vocabulary functional 3D scene graphs and introduces FunGraph3D, and KeySG [38] uses keyframes and hierarchical retrieval to scale scene-graph reasoning to larger environments. These methods emphasize persistent scene memory and object-functional-element relationships, while our target is query-conditioned 3D mask prediction on SceneFun3D and can not directly be address by them.

C Additional Qualitative Results

Fig. 7, Fig. 8 and Fig. 9 show additional qualitative results. Our method outperforms other baselines in various scenes with different functional objects. Specifically, as illustrated in the visual comparisons, UniFunc3D exhibits superior localization accuracy for fine-grained parts, such as cabinet handles and specific drawer sections, where multi-stage baselines like Fun3DU often suffer from over-segmentation. Furthermore, in scenarios involving complex spatial relationships, such as distinguishing between multiple adjacent drawers, our unified reasoning-perception approach accurately identifies the target functional region, whereas training-based methods like AffordBot occasionally fail to generate precise masks for small-scale objects. These results qualitatively validate that our unified architecture not only improves efficiency but also provides more robust and semantically consistent 3D affordance grounding across diverse indoor environments.

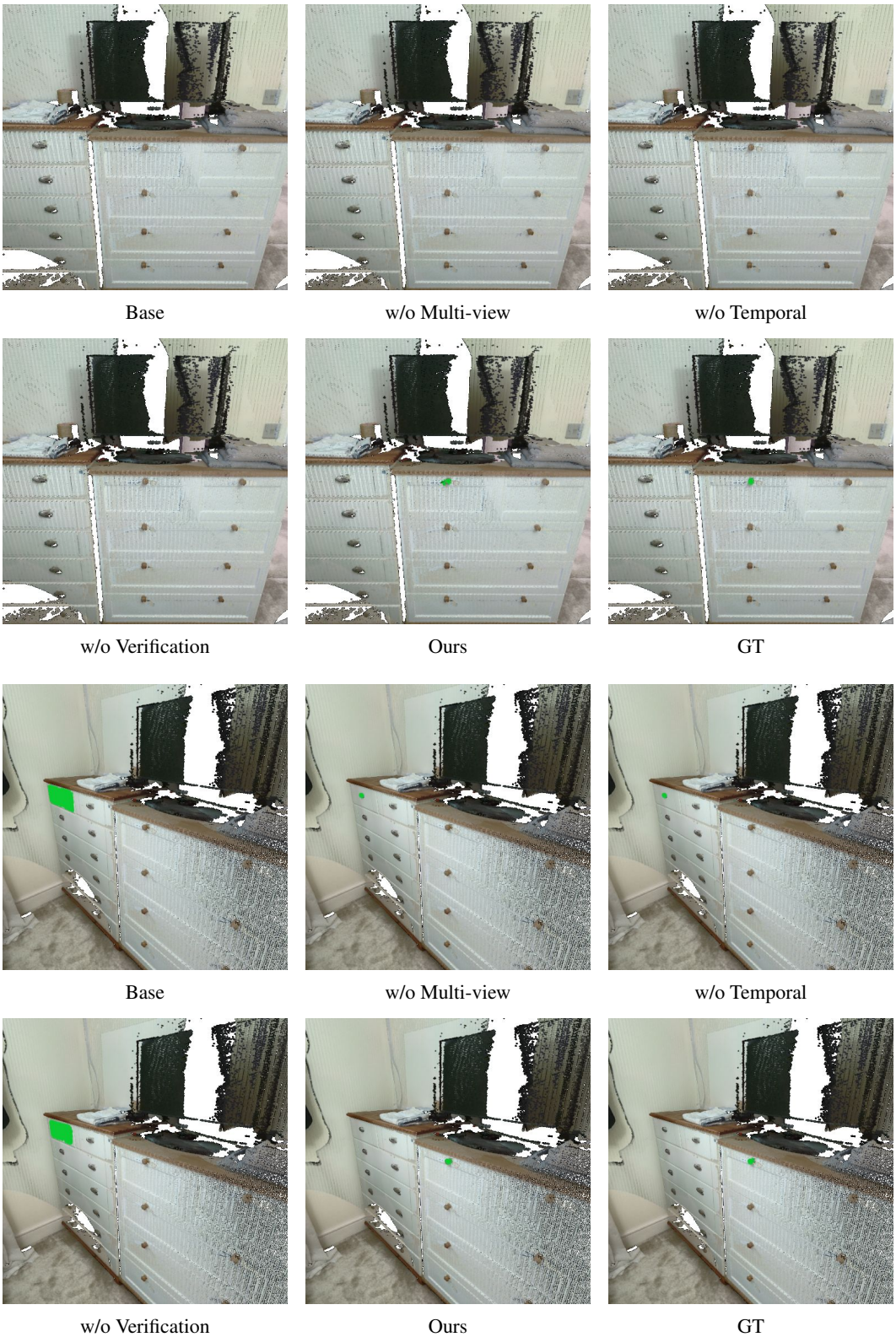


Figure 5: Visual comparisons for ablation study. From left to right, top to bottom: Base, w/o Multi-view, w/o Temporal, w/o Verification, Ours, and GT. Two different viewpoints are shown for the same test example. Input prompt: `Open the top left drawer of the cabinet with the TV on top`.

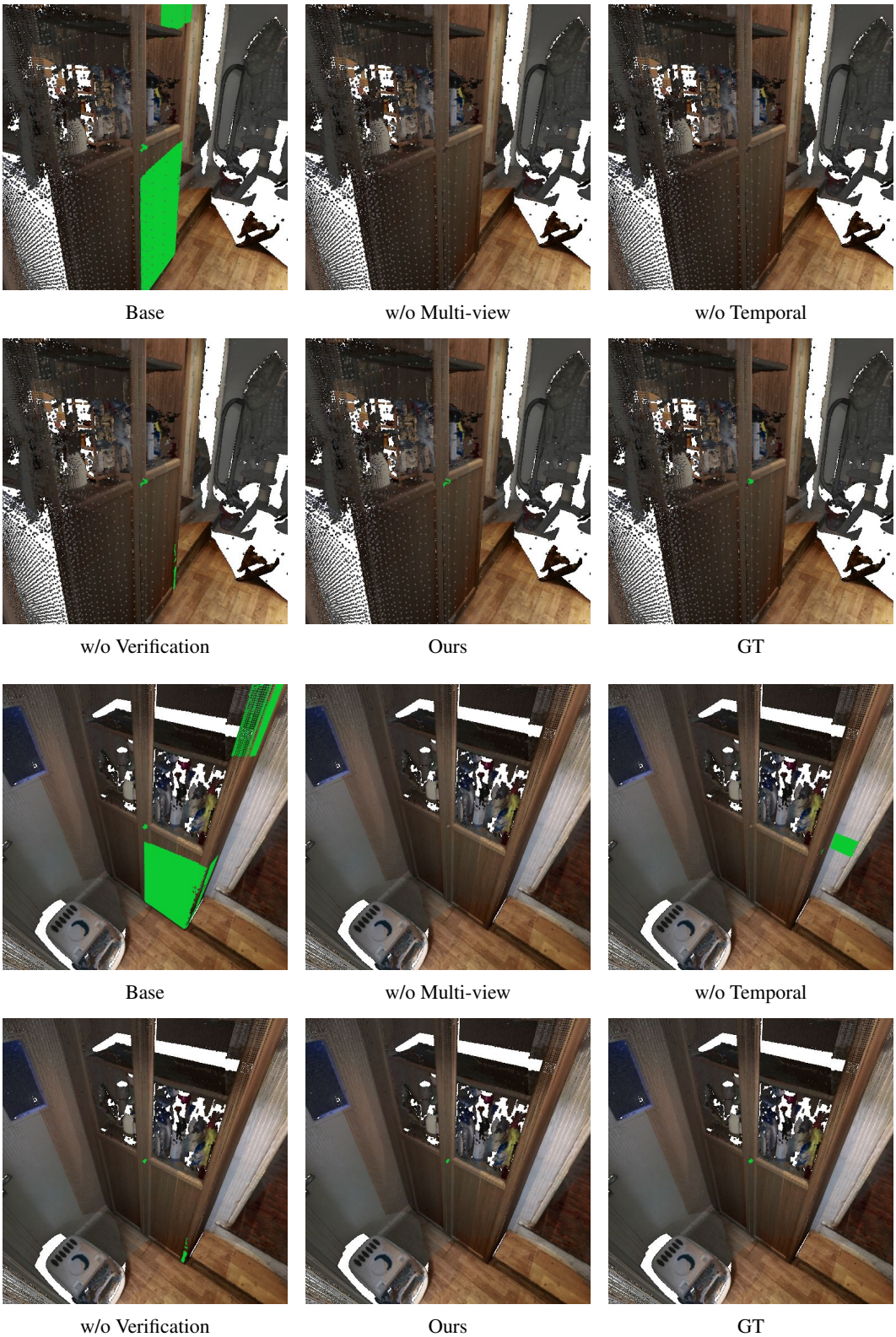


Figure 6: Visual comparisons for ablation study. From left to right, top to bottom: Base, w/o Multi-view, w/o Temporal, w/o Verification, Ours, and GT. Two different viewpoints are shown for the same test example. Input prompt: `Open the right door of the wood and glass display cabinet next to the pet transport box`.

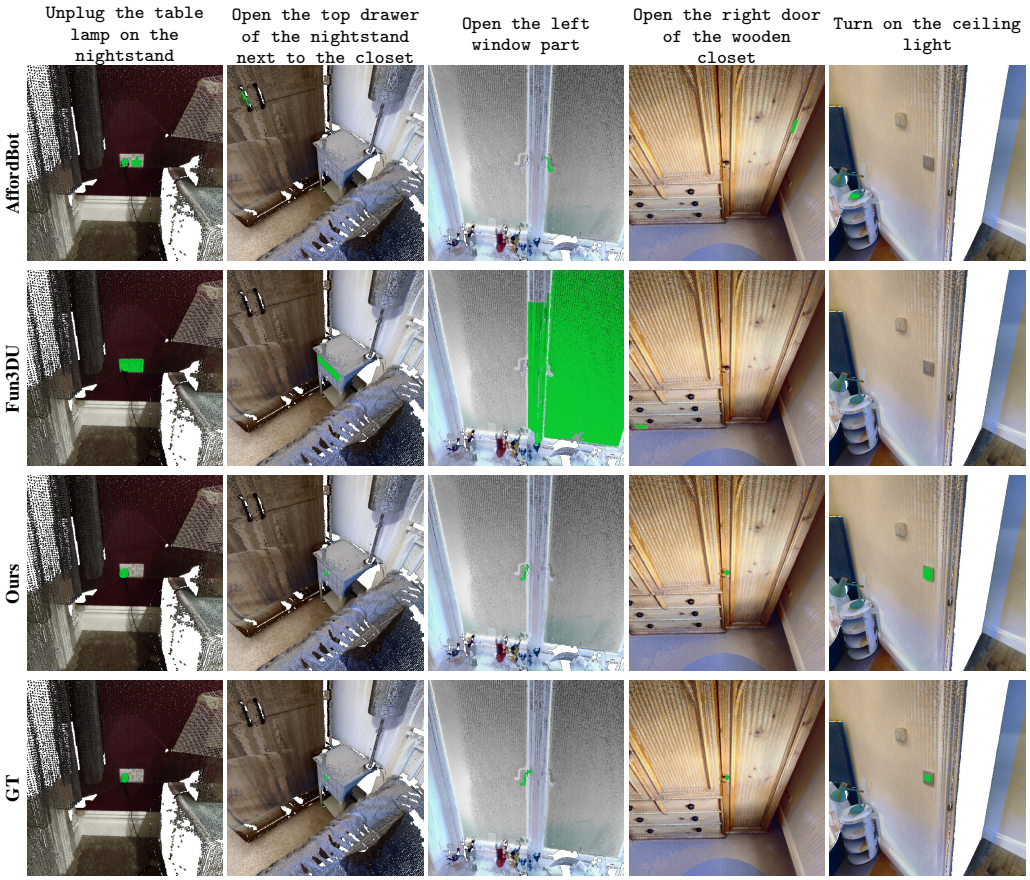


Figure 7: **Qualitative comparison.** We show results for five representative queries (columns) across four methods (rows). GT denotes ground truth.



Figure 8: **Qualitative comparison.** We show results for five representative queries (columns) across four methods (rows). GT denotes ground truth.

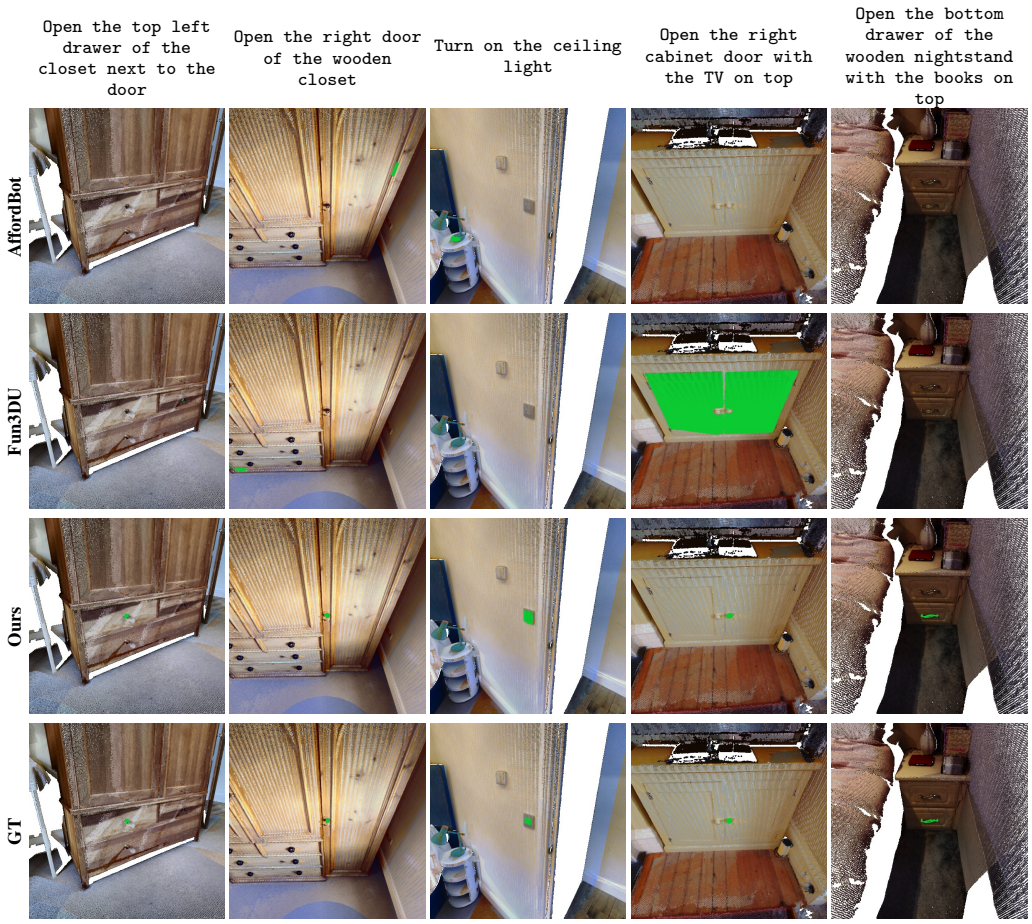


Figure 9: **Qualitative comparison.** We show results for five representative queries (columns) across four methods (rows). GT denotes ground truth.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract claims state-of-the-art performance with a 59.9% relative mIoU improvement on SceneFun3D, and this is directly substantiated by the quantitative results in Tab. 1 (Sec. 4.2) comparing against all prior methods across both evaluation splits.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: A dedicated "Limitations and Future Work" section (Appendix Sec. 4) discusses challenges with extremely small functional parts ($<$0.1% of image area) and severe occlusion, and identifies directions for future work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper presents an empirical systems contribution with no theoretical results, theorems, or formal proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Sec. 4.1 specifies all key hyperparameters (K=4, N=64, resolution 512×384, window radius Δ, SAM3 threshold 0.5, agreement threshold τ=0.7), the backbone models (Qwen3-VL-8B/30B), and the hardware platform (8× H100); Appendix Sec. 3 provides further experimental details.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code will be released on the project page upon acceptance; the evaluation dataset SceneFun3D is publicly available at the benchmark website.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Sec. 4.1 covers dataset splits (split0: 30 val scenes, split1: 200 train scenes), evaluation metrics (AP_{25}, AP_{50}, AR_{25}, AR_{50}, mIoU), and all inference hyperparameters. UniFunc3D is fully training-free, so no optimizer or training details apply.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Error bars are not reported, which is standard practice for benchmark evaluation in 3D scene understanding; the evaluation covers the full test set (split0: 30 scenes, split1: 200 scenes) rather than random subsets, so single-run results are representative.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix Sec. 3 specifies the hardware platform (Intel Xeon Platinum 8468, 8$\times$ NVIDIA H100 GPUs), and Appendix Sec. 1 reports per-scene inference time ($\sim$26 min for UniFunc3D vs. $\sim$82 min for Fun3DU) on this platform.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research involves 3D scene understanding on a publicly released benchmark dataset and does not involve human subjects, sensitive data collection, or harmful applications.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper does not include a broader impact statement. The research advances embodied AI for indoor scene understanding, with clear positive applications in assistive robotics; we see no direct path to significant negative applications beyond general misuse risks common to perception systems.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: UniFunc3D is a training-free inference pipeline that uses publicly available frozen models (Qwen3-VL, SAM3); no new model weights are trained or released, and the evaluation dataset is an existing public benchmark.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All datasets (SceneFun3D), models (Qwen3-VL, SAM3, Qwen2.5-VL), and baselines (Fun3DU, AffordBot, TASA, OpenMask3D, LERF, OpenIns3D) are properly cited with their original papers; all models used are publicly released research models.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [No]

Justification: The paper introduces code for the UniFunc3D inference pipeline, which will be released upon acceptance; documentation is not yet available at submission time.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects; evaluation is conducted entirely on an existing annotated benchmark dataset.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human subjects; no IRB approval is required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: Multimodal LLMs (Qwen3-VL-8B and 30B, and Qwen2.5-VL-72B in ablations) are the central component of UniFunc3D's method; their role in active spatial-temporal grounding, functional object identification, and mask verification is described in detail throughout Sec. 3.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.