JUST-IN-TIME SCENE GRAPH GROWTH: COMBATING PERCEPTUAL SATURATION IN LONG-HORIZON ROBOTICS

Anonymous authors
Paper under double-blind review

ABSTRACT

While 3D Scene Graphs (3DSGs) provide crucial structured representations for embodied agents, conventional Ahead-of-Time, "build-everything-then-filter" pipelines conflict with the real-time, low-latency demands of edge platforms, inducing a *perceptual saturation effect* via severe observation redundancy. To resolve this, we present **JITOMA** (**J**ust-**I**n-**T**ime **O**n-demand **M**emory **A**ctivation), a closed-loop framework that unifies task reasoning, perception, and memory into a just-in-time growth process. Instead of exhaustively mapping the entire environment, JITOMA leverages a top-down task heatmap at the frontend to filter continuous observations, routing minimal streams to maintain a global foundation of low-cost, *dormant anchors*. Upon a cognitive query, the backend Large Language Model (LLM) parses the robotic intent to dynamically awaken task-relevant anchors, triggering expensive semantic operations such as dense node captioning exclusively within the activated local subgraph. To evaluate these dynamic capabilities and study perceptual saturation trade-offs, we introduce **JITOMA-Bench**, a benchmark for long-horizon task switching and complex intent grounding. Across JITOMA-Bench, JITOMA maintains only 1–6 active semantic objects and 0.25–0.28 s/frame across all tiers, showing that semantic computation remains bounded by current task demand rather than accumulated scene complexity.

1 INTRODUCTION

The ability to perceive, memorize, and reason about 3D environments is central to embodied intelligence. 3D Scene Graphs (3DSGs) provide a structured representation for this purpose, encoding objects as nodes and their relationships as edges. With the advent of Vision-Language Models (VLMs), 3DSGs have evolved from closed-set semantics (Hughes et al., 2022; Rosinol et al., 2021; Wu et al., 2021) to open-vocabulary representations (Zhang et al., 2025; Gu et al., 2024; Jatavallabhula et al., 2023; Koch et al., 2024; Linok et al., 2025; Maggio et al., 2024; Werby et al., 2024; Yamazaki et al., 2024; Yan et al., 2025; Chang et al., 2026), enabling zero-shot querying of novel objects. More recently, task-driven (Maggio et al., 2024; 2026; Agia et al., 2022; Maggio & Carlone, 2025) and functional graphs (Zhang et al., 2025; Ju et al., 2025; Büchner et al., 2026; Rotondi et al., 2025) have begun to reshape graph construction around downstream robotic tasks. These advances support increasingly rich spatial memory for robot navigation (Werby et al., 2024; Yan et al., 2025; Gadre et al., 2022; Shah et al., 2023; Yin et al., 2024; Chen et al., 2026; Xia et al., 2026) and manipulation (Yan et al., 2025; Shridhar et al., 2022; Rashid et al., 2023; Honerkamp et al., 2024).

However, most existing 3DSG pipelines still follow an Ahead-of-Time (AOT), "build-everything-then-filter" construction process, instantiating semantic structure before knowing which parts will matter to the downstream task. As shown by our diagnostic study in Sec. 3, this indiscriminate accumulation introduces **observation redundancy** and can lead to a ***perceptual saturation effect***, where additional global semantics no longer improve downstream reasoning and may instead increase semantic clutter and processing cost. Importantly, recent task-driven methods (Maggio et al., 2024; Ju et al., 2025) condition graph structure or granularity on downstream tasks, but operate over task-agnostic primitives or acquired visual observations. Thus, task intent primarily shapes *what* is represented, rather than *where and when* computation is allocated across perception and memory.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Humans offer a useful analogy: rather than resolving every observed object in full detail, perception retains lightweight impressions and allocates fine-grained processing only when a concrete intent demands it. Inspired by this resource-aware behavior, we introduce **just-in-time (JIT) scene graph growth**. Analogous to JIT compilation, semantic structure is instantiated only at runtime where and when it is needed, rather than accumulated globally in advance. This reframes scene graph construction as a process of selective resource allocation: task-irrelevant observations remain lightweight and dormant, while computation is concentrated on task-relevant entities on demand.

Following this principle, we introduce **JITOMA** (**J**ust-**I**n-**T**ime **O**n-demand **M**emory **A**ctivation), an online, closed-loop 3DSG framework that couples task intent with perception and memory. At the frontend, top-down task heatmaps gate continuous observations before memory commitment, while stable observations are retained as lightweight *dormant anchors* without explicit semantic expansion. When a command arrives, JITOMA retrieves and activates only task-relevant anchors, invoking expensive operations such as dense captioning on demand and, when needed, further functional refinement. After execution, reusable semantics are distilled back into dormant memory and the active state collapses. As tasks change, computation therefore shifts between small task-relevant subsets of a persistent memory foundation, rather than accumulating globally over time.

To evaluate these process-level capabilities, we introduce **JITOMA-Bench** (Sec. 5), a three-tier benchmark extending Clio-Bench (Maggio et al., 2024) with continuous real-world trajectories and covering foundational grounding, long-horizon task switching, and complex intent grounding. JITOMA-Bench jointly evaluates grounding accuracy and online efficiency. Across all tiers, JITOMA maintains only 1–6 active semantic objects and 0.25–0.28 s/frame while preserving strong grounding accuracy. This demonstrates that semantic computation can remain bounded by current task demand without sacrificing task-relevant grounding under continual task switching.

2 Related Work

Ahead-of-Time Scene Graph Construction. 3D Scene Graphs (3DSGs) organize sensory observations into structured object nodes and semantic relationships for embodied spatial reasoning (Hughes et al., 2022; Rosinol et al., 2021). Early systems largely relied on closed-set perception (Hughes et al., 2022; Rosinol et al., 2021; Wu et al., 2021), while recent methods increasingly incorporate open-vocabulary semantics through vision-language models (Zhang et al., 2025; Gu et al., 2024; Jatavallabhula et al., 2023; Koch et al., 2024; Linok et al., 2025; Maggio et al., 2024; Werby et al., 2024; Yamazaki et al., 2024; Yan et al., 2025; Chang et al., 2026). Contemporary approaches further enrich scene graphs with hierarchical organization and functional knowledge to support downstream interaction (Zhang et al., 2025; Ju et al., 2025; Büchner et al., 2026). Despite these advances in representation, most pipelines still follow an Ahead-of-Time (AOT) construction pattern: visual evidence and semantic structure are progressively instantiated during mapping before their relevance to a future task is known. This design provides rich persistent memory, but allocates perception and semantic computation broadly across the scene. As exploration grows, task-irrelevant observations and semantics therefore accumulate together with useful content, creating the observation redundancy and perceptual saturation studied in this work.

Task-Driven Representations and Just-In-Time Growth. A growing line of work instead conditions scene representations on downstream tasks. Taskography sparsifies existing 3DSGs for tractable task planning (Agia et al., 2022); Clio clusters task-agnostic 3D primitives into task-relevant objects and regions (Maggio et al., 2024); Bayesian Fields introduces task-dependent granularity into open-set semantic mapping (Maggio & Carlone, 2025); and MomaGraph constructs task-oriented graph structure from acquired visual observations (Ju et al., 2025). These methods demonstrate the benefit of adapting graph content or granularity to downstream requirements, but task conditioning generally operates after visual evidence or low-level primitives have already been acquired. FOUND-IT (Maggio et al., 2026) moves further toward on-demand construction by retaining a visual memory and resolving object and region granularity when runtime tasks arrive. JITOMA builds on this direction from a complementary perspective: task intent is used not only to determine *what* should be represented, but also to govern *where and when* computation is allocated across perception and memory throughout online scene construction.

3 MOTIVATION: PERCEPTUAL SATURATION IN AOT SCENE GRAPHS

Conventional Ahead-of-Time (AOT) scene graph construction pursues increasingly complete global representations. We ask whether this exhaustive accumulation is always beneficial for reasoning, or whether excessive task-irrelevant semantics can instead become a source of cognitive burden.

Does global structural completeness overwhelm downstream reasoners? We conduct a diagnostic study following the complex-intent setting of Tier 3 in JITOMA-Bench (Sec. 5). A Large Language Model (Qwen3.5-9b (Qwen Team, 2026)) plans the command "pack my eyeglasses for travel" in a cluttered real-world cubicle. We compare two upstream representations: **AOT Full Graph**, containing all 18 environmental entities constructed by a task-agnostic pipeline, and **JIT Subgraph (Ours)**, containing only the 3 task-relevant entities (*eyeglasses*, *glasses case*, and *backpacks*) activated on demand. Both conditions use the identical planner and prompt, differing only in the scene-graph entities exposed to the planner (Appendix A.2.1).

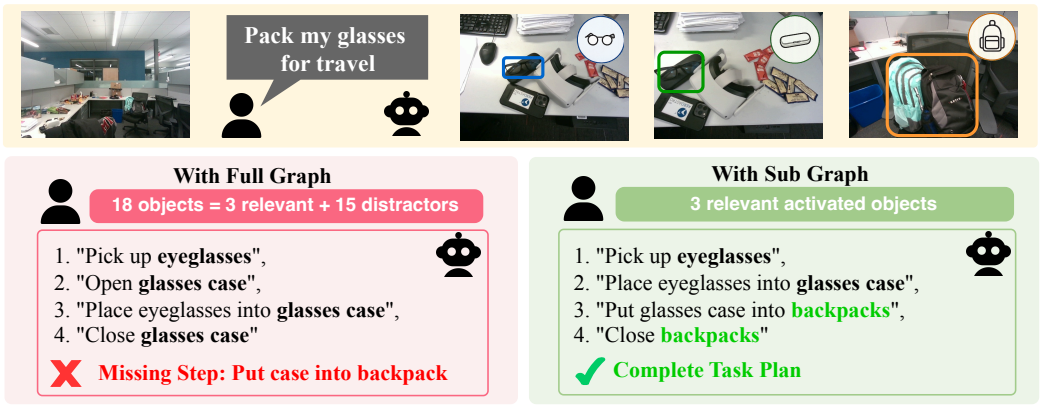

Figure 1: **Illustrative example of perceptual saturation. Left:** AOT exposes all 18 entities, including 15 distractors, causing the planner to miss the final containment step. **Right:** JIT exposes only 3 task-relevant entities, enabling the same planner to recover the complete execution sequence.

As shown in Fig. 1, the AOT graph exposes the planner to 15 task-irrelevant distractors (e.g., *hardware drills* and *mudstone rocks*). The planner terminates after placing the eyeglasses into the case and omits the final step of placing the case into the backpack, whereas the same planner produces the complete sequence when given only the three task-relevant entities.

To examine whether this failure mode extends beyond a single example, we further conduct a controlled distractor study over all 25 Tier 3 tasks (Appendix A.2.2). Holding the task-relevant graph content and planner prompt fixed, we progressively add $k \in \{0, 5, 10, 15\}$ task-irrelevant entities. Across 1,194 planner requests, mean all-target coverage decreases from 82.31% to 75.24%, while latent-target coverage drops more substantially from 68.10% to 56.30%. In contrast, primary-target coverage remains nearly unchanged (98.52% vs. 98.11%), suggesting that irrelevant semantic load primarily disrupts the recovery of implicit objects required for task execution.

This observation motivates controlling semantic load before it reaches the downstream reasoner. While task-driven filtering can mitigate clutter at query time, JITOMA moves this selectivity upstream: task intent governs where perceptual and semantic computation are allocated during scene construction. Taken together, the qualitative example and controlled study support a ***perceptual saturation effect***: when task-relevant information is held fixed, increasing irrelevant semantic load can impair the recovery of task-required information, particularly latent dependencies.

Finding: Perceptual Saturation Effect

With task-relevant graph content held fixed, increasing task-irrelevant semantic load can reduce downstream target coverage, with the strongest degradation on latent objects required for task completion.

4 METHOD

4.1 PROBLEM FORMULATION AND ARCHITECTURE OVERVIEW

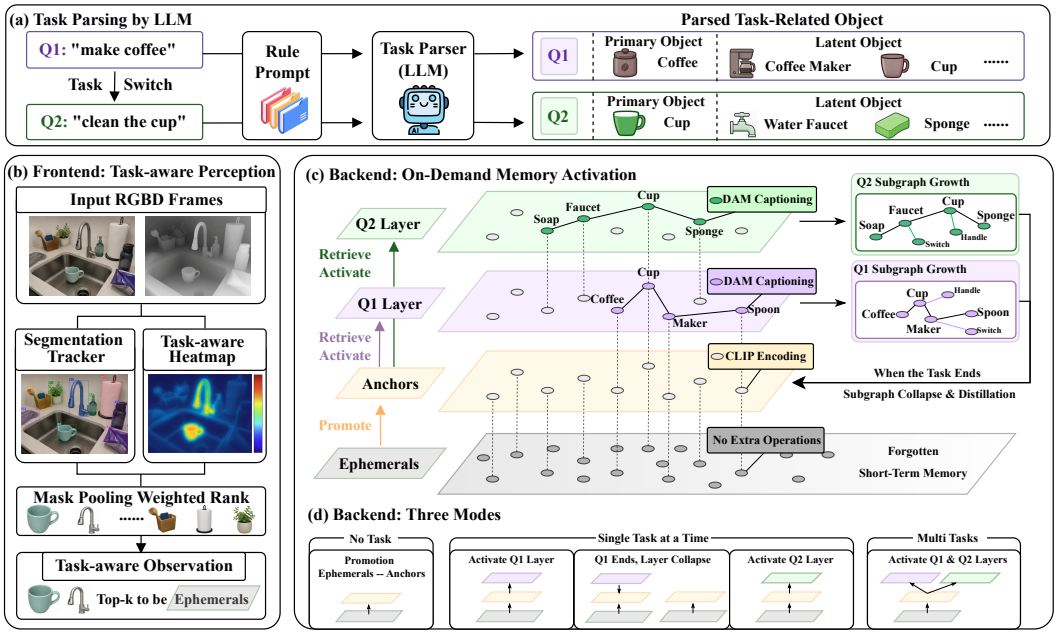

Figure 2: **Overview of JITOMA. (a) Intent Parsing:** An LLM extracts primary and latent targets. **(b) Frontend Perception:** Task heatmaps gate RGB-D observations before memory commitment. **(c) Backend JIT Activation:** Dormant anchors are selectively activated for semantic expansion and optional functional refinement. **(d) Operating Modes:** JITOMA supports task-free writing, single-query activation, and concurrent queries over a shared dormant memory.

JITOMA targets online 3D scene graph construction under long-horizon embodied execution, where conventional Ahead-of-Time (AOT) pipelines instantiate global semantic structures before knowing which parts of the scene will be used. Given a continuous RGB-D stream and camera poses $\{(I_t, D_t, P_t)\}_{t=1}^{T}$, JITOMA instead treats graph construction as a *just-in-time growth process*: exploration maintains a sparse memory foundation, while expensive semantic computation is triggered only when required by the current task. At time t, JITOMA maintains a two-tier memory state

$$\mathcal{M}_t = \left(V_t^{\text{eph}}, V_t^{\text{anc}}\right), \tag{1}$$

where V_t^{eph} contains ephemeral short-term hypotheses and V_t^{anc} contains long-lived dormant anchors. Ephemeral hypotheses represent transient observations, whereas dormant anchors retain stable observational evidence without explicit semantics such as object captions. Upon a command q, JITOMA activates only task-relevant anchors, materializes their semantic structure on demand, and distills reusable semantics back into dormant memory after execution. Thus, expensive graph structure follows active cognitive demand rather than accumulating with exploration time.

4.2 TASK-GUIDED PERCEPTION AND MEMORY FORMATION

Unlike task-driven representations operating on acquired evidence, JITOMA lets task intent govern memory formation before retrieval. Given command q, an LLM parser extracts $C(q) = C_{\text{pri}}(q) \cup C_{\text{lat}}(q)$, where C_{pri} denotes explicit targets and C_{lat} execution-implied objects. For example, Fig. 2(a) parses "clean the cup" into $C_{\text{pri}} = \{cup\}$ and $C_{\text{lat}} = \{water\ faucet, sponge\}$.

At timestamp t, a class-agnostic segmentation tracker (Gorlo et al., 2026) produces $O_t = \{o_{t,i}\}_{i=1}^{N_t}$, with $o_{t,i} = (M_{t,i}, x_{t,i}, g_{t,i})$ denoting its 2D mask, crop, and coarse 3D geometry. Before semantic labeling, the parsed concepts produce a top-down task heatmap H_t^q through a vision-language

alignment model (Lüddecke & Ecker, 2022), and only the most relevant observations are admitted:

$$\widetilde{O}_t(q) = \mathrm{TopK}_{o_{t,i} \in O_t} \mathrm{Pool}(H_t^q \odot M_{t,i}) \,. \tag{2}$$

This gate acts before memory commitment, suppressing redundant observations rather than filtering a fully constructed graph. When no task is active, JITOMA uses conservative writing, admitting new tracks and retaining repeated observations only when visual evidence improves (Appendix A.3.2).

From Ephemeral Hypotheses to Dormant Anchors. Admitted observations $\widetilde{O}_t(q)$ enter the *Ephemerals* layer as short-term hypotheses and accumulate lightweight visual and geometric evidence. A hypothesis is promoted to a dormant anchor $v_j \in V_t^{\mathrm{anc}}$ after sufficient observations; otherwise, it expires. At promotion, JITOMA stores a representative crop $\bar{x}_j$ and visual key $z_j = \mathrm{CLIP}_{\mathrm{img}}(\bar{x}_j)$. Anchors retain stable evidence but remain semantically dormant, so persistent memory grows without globally materializing object semantics.

4.3 On-Demand Memory Activation

When a command q arrives, JITOMA activates memory through a lightweight retrieval stage followed by selective semantic verification. For readability, we omit the time subscript and operate on the current dormant anchor set V^{anc}. For each concept $c \in C(q)$, candidate anchors are retrieved using the CLIP visual key together with accumulated frontend heatmap evidence:

$$B(c) = \mathrm{TopK}^{K_{\mathrm{act}}}_{v_j \in V^{\mathrm{anc}}} \left[\cos(\mathrm{CLIP}_{\mathrm{text}}(c), z_j) + \lambda \tilde{h}_j^c \right] . \tag{3}$$

Here, $\tilde{h}_j^c$ denotes the accumulated task-heatmap evidence associated with anchor v_j. Only anchors in $B(c)$ invoke expensive semantic processing; all others remain dormant. JITOMA batch-captions the retrieved candidates with DAM (Gorlo et al., 2026; Lian et al., 2025), producing descriptions d_j, and then uses an LLM reranker to select the best anchor for each concept:

$$V^{\mathrm{act}}(q) = \left\{ \mathrm{LLM}_{\mathrm{rerank}}\big(q, c, \{(v_j, d_j)\}_{v_j \in B(c)}\big) \;\middle|\; c \in C(q) \right\} \subseteq V^{\mathrm{anc}}. \tag{4}$$

Unselected candidates return immediately to the dormant state. Thus, captioning and language reasoning are restricted to a small task-relevant candidate set instead of being applied globally.

4.4 Just-In-Time Semantic Growth

Semantic growth occurs only after activation: selected anchors $V^{\mathrm{act}}(q)$ materialize explicit object semantics d_j, while the rest of memory remains dormant. Thus, JITOMA expands only task-relevant objects, confining semantic computation to current task demand.

Optional functional refinement. Beyond the object-level semantic activation evaluated in this work, JITOMA can optionally refine an activated object with a task-conditioned interaction part when finer-grained physical interaction is required. Given an activated anchor v_j, its representative crop $\bar{x}_j$, caption d_j, and command q, a VLM predicts the interaction-relevant part, which is grounded in 2D and back-projected into 3D:

$$\ell_j^q = \mathrm{VLM}_{\mathrm{part}}(q, \bar{x}_j, d_j), \qquad b_j^q = \mathrm{OBB}\big(\Pi^{-1}\big(\mathrm{Seg}(\bar{x}_j, \ell_j^q), D_j, P_j\big)\big) \,. \tag{5}$$

Here, ℓ_j^q denotes the interaction-relevant part and b_j^q its localized 3D extent; the resulting node $p_j^q = (b_j^q, \ell_j^q)$ is attached only to the activated anchor. This optional extension applies the JIT principle to finer-grained structure, while our evaluation focuses on semantic activation and grounding.

4.5 Semantic Distillation and Operating Modes

After task execution, JITOMA collapses the active semantic state and distills reusable outputs into the corresponding dormant anchors. For each $v_j \in V^{\mathrm{act}}(q)$, its caption d_j and, when available, functional refinement p_j^q are retained as auxiliary attributes via $\mathcal{A}_j \leftarrow \mathcal{A}_j \cup \{d_j, p_j^q\}$. Temporary heatmaps, masks, retrieval candidates, and task-specific overlays are discarded, while distilled semantics remain dormant until reactivated by a subsequent query.

As illustrated in Fig. 2(d), JITOMA supports three modes. With no query, it performs conservative writing and anchor promotion. With a single query, relevant anchors are activated, semantically expanded, distilled after execution, and returned to the dormant state. With concurrent queries, multiple task-specific overlays share the same dormant anchor foundation.

5 Dataset and Benchmark

Clio-Bench (Maggio et al., 2024) provides real-world RGB-D trajectories and geometric ground truth for evaluating task-driven open-set 3DSGs. However, existing benchmarks overlook three aspects: (G1) **Peak Active-State Dynamics**, as final graph statistics do not capture online peak semantic-state growth and computation; (G2) **Long-Horizon Multi-Tasking**, as evaluation rarely considers continuous task switching within the same scene; and (G3) **Complex Intent Grounding**, as commands involve only explicit target objects rather than multiple primary and latent entities.

To address these gaps, we introduce **JITOMA-Bench**, extending Clio's three scenes with a three-tier evaluation protocol covering single-query grounding, multi-task adaptation, and complex intent grounding (Sec. 5.1). JITOMA-Bench contains 126 evaluation tasks, 156 task-target references, and 80 unique annotated 3D boxes, with detailed statistics provided in Appendix A.1.

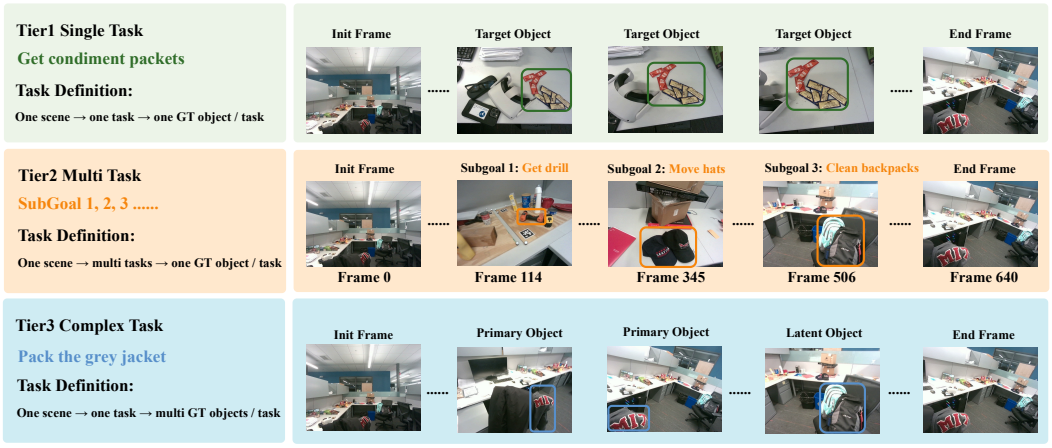

Figure 3: **Overview of the three-tier JITOMA-Bench evaluation.** The benchmark progressively evaluates explicit single-object grounding (Tier 1), long-horizon task switching within a continuous scene (Tier 2), and complex instruction grounding over multiple primary and latent objects (Tier 3).

5.1 Task Annotations and Complementary Evaluation Tracks

Tier 1: Foundational Grounding. As shown in Fig. 3, Tier 1 evaluates each explicit single-GT task independently, providing a controlled measure of conventional 3DSG object grounding.

Tier 2: Long-Horizon Dynamics. Tier 2 addresses G2 by evaluating continual adaptation under an evolving task stream within the same scene. To construct potentially overlapping tasks, we manually annotate the first- and last-visible frames of each target (e.g., Frame X for Subgoal x in Fig. 3). Each task is activated at its first-visible frame and evaluated ten frames after its last-visible frame.

Tier 3: Complex Intent Grounding. Tier 3 evaluates instructions involving multiple explicit and latent objects following the Tier 1 timing protocol. Using Clio ground-truth labels, GPT-5.4 (OpenAI, 2026) generates candidate tasks for the Apartment, Office, and Cubicle scenes by composing single-object tasks into multi-object instructions. The authors assign primary and latent objects according to Clio labels, and three PhD researchers independently review each instruction. Candidates are removed if at least two reviewers identify ambiguity or unnecessary objects for task completion.

5.2 Evaluation Metrics

Our evaluation follows two complementary objectives: **grounding accuracy** and **system efficiency**. For accuracy, we report the top-1 Intersection over Union (IoU), averaged across queries, to measure geometric agreement between the retrieved object and its ground truth. We further define mR@K as the arithmetic mean of Recall@K over IoU thresholds $\{0.1, 0.2, 0.3\}$, capturing both semantic retrieval and localization robustness. In Tier 3, Primary and Latent concepts are evaluated sepa-

rately before being averaged within each task, ensuring that performance reflects both explicit-object grounding and implicit-intent understanding.

For efficiency, we distinguish the *active semantic state* from the persistent visual memory. **Objs** counts the semantic object nodes active at query time, while **Peak** is the maximum active semantic-node count over the scene replay. For JITOMA, dormant anchors are excluded from these active-state counts and reported separately in parentheses; methods without an explicit dormant/active split count all nodes in their native semantic representation. Thus, Objs and Peak characterize semantic working-set complexity rather than byte-level memory footprint. Finally, **TPF** (Time Per Frame) measures the average processing time per frame.

6 EXPERIMENTS

We evaluate JITOMA on JITOMA-Bench (Sec. 5), covering single-query grounding, long-horizon task switching, and complex Primary–Latent intent grounding. We compare JITOMA against open-vocabulary and task-driven 3DSG baselines under both grounding accuracy and efficiency metrics.

6.1 EXPERIMENTAL SETUP

Baselines. We benchmark JITOMA against three representative 3DSG frameworks: **ConceptGraphs** (Gu et al., 2024), an open-vocabulary scene graph method that constructs a global semantic representation by aggregating object-level features; **ReasoningGraph** (Puigjaner et al., 2026), a hierarchical scene graph framework that augments object representations with relational information for structured scene understanding; and **Clio** (Maggio et al., 2024), a real-time task-driven scene graph method that derives task-relevant representations from open-world 3D primitives.

Implementation Details. We implement JITOMA using Qwen3.5-9b (Qwen Team, 2026) as the foundation model, paired with DAM (Lian et al., 2025) in batch mode (Gorlo et al., 2026) for on-demand captioning. All implementation details and configurations are provided in Appendix A.3.

Table 1: **Evaluation on JITOMA-Bench.** We compare JITOMA with sota baselines across all three evaluation tiers, measuring grounding accuracy and efficiency. IoU, mR@1, and mR@3 measure grounding accuracy, while Objs, Peak, and TPF measure query-time active semantic size, peak active semantic size, and per-frame latency. For JITOMA, Peak is reported as active(dormant), explicitly exposing the retained dormant-anchor state. For Tier 3, baseline accuracy is reported as **Vanilla / Augmented**, where Vanilla uses each method's original task interpretation pipeline and Augmented replaces task parsing with our LLM-based parser to isolate retrieval capability from intent parsing.

Scene	Method	Tier 1: Foundational Single-Goal						Tier 2: Long-Horizon Temporal						Tier 3: Complex Intent					
		Accuracy ↑			*Efficiency*			*Accuracy* ↑			*Efficiency*			*Accuracy* ↑			*Efficiency*		
		IoU	mR@1	mR@3	Objs ↓	Peak ↓	TPF ↓	IoU	mR@1	mR@3	Objs ↓	Peak ↓	TPF ↓	IoU	mR@1	mR@3	Objs ↓	Peak ↓	TPF ↓
Apartment	ConceptGraphs	10.9	20.5	28.2	444	1479	5.22	17.6	33.3	**52.8**	452	1476	6.03	2.9 / 14.3	7.4 / 25.9	7.4 / 35.2	451	1494	5.26
	ReasoningGraph	9.7	25.6	26.9	73	488	2.91	12.6	33.3	33.3	105	488	2.91	4.4 / 6.4	11.1 / 16.7	11.1 / 16.7	73	488	2.91
	Clio	9.5	19.2	49.3	29	717	0.27	10.9	25.0	27.8	4	637	0.35	5.6 / 9.5	12.5 / 21.5	16.7 / 25.7	30	704	0.27
	JITOMA (Ours)	**16.2**	**32.1**	**51.3**	2	4(488)	**0.25**	**21.4**	**47.2**	**52.8**	2	4(494)	**0.26**	**14.8**	**27.0**	**35.4**	6	12(467)	**0.26**
Office	ConceptGraphs	13.5	31.8	37.9	926	5793	11.10	19.0	40.5	45.2	924	5796	12.58	1.8 / 13.3	0.0 / 35.7	4.8 / 47.6	931	5797	12.13
	ReasoningGraph	8.4	21.2	36.4	35	605	3.04	17.3	42.9	45.2	102	605	3.04	10.2 / 9.2	26.2 / 23.8	26.2 / 47.6	35	605	3.04
	Clio	17.2	39.4	56.0	19	723	0.28	17.7	38.1	47.6	5	678	0.42	1.7 / 16.5	5.6 / 40.7	5.6 / 50.0	19	619	0.29
	JITOMA (Ours)	**20.4**	**53.0**	**60.6**	1	3(510)	**0.25**	**29.1**	**64.3**	**73.8**	1	4(512)	**0.26**	**19.8**	**50.7**	**58.0**	6	12(512)	**0.26**
Cubicle	ConceptGraphs	11.2	24.1	38.9	235	758	5.18	11.3	22.2	63.0	224	745	5.37	3.7 / 6.4	4.4 / 5.8	15.9 / 31.9	237	766	5.32
	ReasoningGraph	8.4	18.5	24.1	89	260	2.92	12.9	22.2	29.6	124	260	2.92	3.6 / 9.3	10.1 / 18.8	15.9 / 29.0	89	260	2.92
	Clio	19.3	48.1	59.3	41	370	0.33	**23.8**	**66.7**	**74.1**	5	408	0.46	7.6 / 17.3	18.2 / 44.4	21.2 / **57.6**	44	460	0.33
	JITOMA (Ours)	**21.9**	**55.6**	**66.7**	2	4(284)	**0.26**	22.4	**66.7**	**74.1**	2	4(289)	**0.28**	**21.6**	**45.2**	**57.6**	6	12(288)	**0.28**

6.2 MAIN RESULTS: ACCURACY AND SYSTEM EFFICIENCY

We report the overall evaluation on JITOMA-Bench in Table 1. JITOMA achieves strong grounding accuracy while maintaining a compact active representation and stable online processing cost across increasingly challenging task settings.

Grounding accuracy under diverse task settings. JITOMA achieves strong grounding performance across all three evaluation tiers, demonstrating robustness from single-object retrieval to complex Primary–Latent intent grounding. On Tier 1, JITOMA obtains the best overall accuracy

in most scenes, e.g., 20.4 IoU / 53.0 mR@1 / 60.6 mR@3 in Office and 21.9 IoU / 55.6 mR@1 / 66.7 mR@3 in Cubicle. On Tier 2, it maintains high retrieval accuracy under long-horizon task streams, achieving 29.1 IoU / 64.3 mR@1 / 73.8 mR@3 in Office. For Tier 3, where instructions require both explicit and latent object grounding, JITOMA achieves the highest accuracy across all scenes, reaching 19.8 IoU / 50.7 mR@1 / 58.0 mR@3 in Office. These results suggest that task-conditioned memory formation helps reduce irrelevant observations before semantic expansion, enabling retrieval from a more compact task-relevant representation.

Bounded semantic growth and online efficiency. Table 1 separates JITOMA's active semantic state from its dormant visual memory: Objs/Peak count active semantic nodes, while parenthesized Peak values expose the retained dormant anchors; Table 2 further separates captioned and CLIP-resident states. Across Tier 1/2, JITOMA uses only 1–2 active objects at query time (peak ≤ 4) while retaining 284–512 dormant anchors; Tier 3 uses six active objects with a peak of twelve. Thus, our claim concerns bounded semantic materialization rather than negligible total storage. Despite invoking on-demand dense captioning, JITOMA maintains 0.25–0.28 s/frame, at or below Clio's 0.27–0.46 s/frame across all settings. This shows that JIT allocation preserves a persistent visual memory while avoiding global semantic expansion.

6.3 Component Ablation

We ablate the two mechanisms that define JITOMA's just-in-time growth process: *task-aware observation admission* at the perception frontend and *on-demand semantic activation* at the memory backend. The former determines which observations are committed to memory, whereas the latter delays expensive semantic expansion until an object becomes relevant to the current task. We conduct the study on Tier 2 of JITOMA-Bench, which evaluates continual grounding under a stream of task switches with persistent scene memory. The default hyperparameters are used throughout this analysis, and their sensitivity is further studied in Appendix A.3.3.

Ablation design. We decompose JITOMA along its two AOT–JIT transitions: observation admission at the frontend and semantic expansion at the backend. At the frontend, AOT admits observations independently of the task, whereas JIT performs task-guided admission; at the backend, AOT expands all objects proactively, whereas JIT activates semantics on demand. Variants A, B, C, and E span the resulting 2×2 design, with E denoting the full JITOMA. Variant D† is a task-independent control that matches E's admission budget while selecting observations without task, isolating task relevance from sparsity. Results are averaged over the three Tier 2 scenes and summarized in Table 2.

Table 2: Component ablation on Tier 2 of JITOMA-Bench. AOT/JIT denote task-independent/task-guided frontend admission and proactive/on-demand backend expansion. † matches JITOMA's per-frame admission budget. Cap. and CLIP denote captioned and CLIP-resident objects.

	Configuration		Accuracy ↑			Efficiency ↓				
Variant	Frontend	Backend	IoU	mR@1	mR@3	Objs_{Cap}	$\text{Objs}_{\text{CLIP}}$	Peak_{Cap}	$\text{Peak}_{\text{CLIP}}$	TPF
A	AOT	AOT	12.8	31.0	45.9	355.0	246.9	512.0	387.0	0.460
B	JIT	AOT	22.9	52.4	64.7	223.4	170.8	421.0	331.3	0.436
C	AOT	JIT	13.5	30.4	45.9	31.4	247.1	49.7	387.0	0.336
D†	AOT†	JIT	13.7	31.9	42.6	31.0	208.1	51.0	368.3	0.311
E (Ours)	**JIT**	**JIT**	**24.3**	**57.5**	**67.1**	**28.5**	**170.5**	**47.0**	**330.7**	**0.282**

Task-aware perception limits memory accumulation. Comparing A and B isolates the frontend JIT transition. Task-guided admission reduces the average CLIP-resident object count from 246.9 to 170.8 and its peak from 387.0 to 331.3, while lowering TPF from 0.460 to 0.436 s/frame and improving IoU from 12.8 to 22.9 and mR@1 from 31.0 to 52.4. Thus, frontend JIT reduces the visual state committed to persistent memory before semantic processing.

On-demand activation limits semantic computation. Comparing A and C isolates the backend transition. With nearly unchanged CLIP-resident memory, JIT activation reduces the average number of captioned objects from 355.0 to 31.4 and the peak from 512.0 to 49.7, while lowering TPF

from 0.460 to 0.336 s/frame. Thus, backend JIT primarily avoids proactive semantic materialization rather than eliminating the dormant visual memory itself.

Complementary frontend and backend resource allocation. The two JIT transitions reduce different resource costs. Frontend task-aware admission controls *what enters memory*, limiting persistent visual-state growth and downstream processing. Backend on-demand activation controls *when semantic computation is incurred*, reducing captioned objects from 355.0 to 31.4 and TPF from 0.460 to 0.336 s/frame while leaving CLIP-resident memory nearly unchanged. Together, JITOMA limits persistent memory growth at the frontend and semantic compute at the backend.

6.4 Qualitative Analysis of Just-In-Time Memory Growth

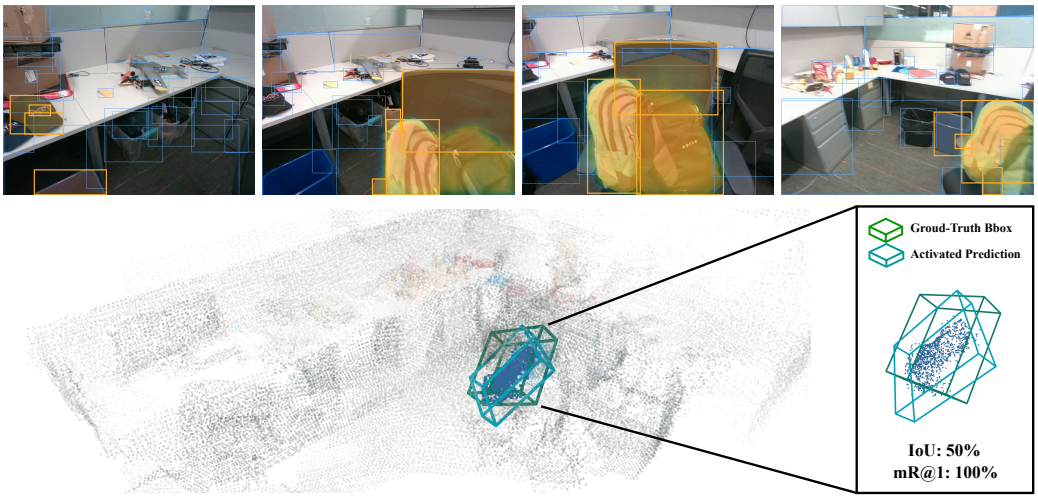

Figure 4: **Qualitative example of just-in-time memory growth.** The example illustrates JITOMA's task-guided perception and on-demand semantic activation under the "Get backpacks" command. **Top:** During frontend perception, blue boxes denote tracked objects that are not admitted because they provide neither task relevance nor novel visual evidence. Orange boxes denote newly admitted observations, where heatmap-highlighted regions correspond to task-relevant objects and non-highlighted regions correspond to novel but task-irrelevant objects. **Bottom:** JITOMA activates only the task-relevant anchor from the dormant memory foundation. The activated prediction (cyan) is compared with the ground-truth bounding box (green), achieving IoU=50% and mR@1=100%.

Fig. 4 illustrates how JITOMA allocates perception and memory resources under the "Get backpacks" command. During frontend perception, previously observed but irrelevant objects (blue boxes) are suppressed without forming new memory hypotheses, while novel observations (orange boxes) are selectively admitted according to task relevance and visual novelty. After activation, JITOMA expands only the task-relevant dormant anchors, producing a compact semantic representation whose activated prediction aligns with the ground-truth object. This example demonstrates that JITOMA separates persistent geometric memory from task-specific semantic growth, enabling detailed semantic processing only when required by the current task.

7 Conclusion

We introduced **JITOMA**, a just-in-time scene graph framework that shifts 3D memory construction from exhaustive Ahead-of-Time accumulation to task-conditioned on-demand growth. By maintaining lightweight dormant anchors and activating semantic structures only when required, JITOMA achieves efficient embodied scene understanding while preserving grounding capability under increasingly complex tasks. Through JITOMA-Bench and extensive experiments, we demonstrate that controlling not only *what* a scene graph represents but also *when and where* its semantic structure is instantiated is essential for scalable robotic perception.

AI USE STATEMENT

We used generative AI tools in three supporting roles during the preparation of this work. First, AI-assisted writing tools were used to improve grammar, clarity, and presentation of author-written text. Second, generative AI was used to propose candidate task annotations and instructions for JITOMA-Bench; all generated candidates were subsequently reviewed, corrected when necessary, and approved by human annotators according to the annotation protocol described in the paper. Third, generative AI was used to generate preliminary visual concepts and layout ideas that assisted the authors in designing several figures; the final figures and scientific content were produced and verified by the authors. Generative AI was not used to fabricate experimental measurements or results, replace human validation of benchmark annotations, or make final scientific judgments regarding the methods, analyses, or conclusions of this work. All AI-assisted text, annotations, and visual materials were manually reviewed for correctness, consistency, and appropriate attribution. The authors take full responsibility for the final content of this paper, including all text, claims, data, experiments, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide sufficient details to reproduce JITOMA and JITOMA-Bench. The JITOMA pipeline is specified in Sec. 4, while benchmark construction, task annotations, and evaluation metrics are described in Sec. 5, with dataset statistics and annotation details in Appendix A.1. The controlled perceptual-saturation study, including prompt construction, distractor controls, and evaluation protocol, is given in Appendix A.2.2. Implementation settings, memory-management rules, default hyperparameters, and sensitivity analysis are provided in Appendix A.3, and experimental baselines and model configurations in Sec. 6. Upon acceptance, we will release the implementation, benchmark annotations, and evaluation scripts to facilitate reproducibility and future research.

REFERENCES

Christopher Agia, Krishna Murthy Jatavallabhula, Mohamed Khodeir, Ondrej Miksik, Vibhav Vineet, Mustafa Mukadam, Liam Paull, and Florian Shkurti. Taskography: Evaluating robot task planning over large 3d scene graphs. In *Conference on Robot Learning*, pp. 46–58. PMLR, 2022.

Martin Büchner, Adrian Röfer, Tim Engelbracht, Tim Welschehold, Zuria Bauer, Hermann Blum, Marc Pollefeys, and Abhinav Valada. Articulated 3d scene graphs for open-world mobile manipulation. *arXiv preprint arXiv:2602.16356*, 2026.

Yue Chang, Rufeng Chen, Zhaofan Zhang, Yi Chen, Yifan Tian, and Sihong Xie. Rag-3dsg: Enhancing 3d scene graphs with re-shot guided retrieval-augmented generation. *arXiv preprint arXiv:2601.10168*, 2026.

Rufeng Chen, Yue Chang, Xiaqiang Tang, Hechang Chen, and Sihong Xie. Psg-nav: Probabilistic scene graph navigation via multiverse decision making. *arXiv preprint arXiv:2606.01313*, 2026.

Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Clip on wheels: Zero-shot object navigation as object localization and exploration. *arXiv preprint arXiv:2203.10421*, 3(4):7, 2022.

Nicolas Gorlo, Lukas Schmid, and Luca Carlone. Describe anything anywhere at any moment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 35002–35013, 2026.

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

Daniel Honerkamp, Martin Büchner, Fabien Despinoy, Tim Welschehold, and Abhinav Valada. Language-grounded dynamic scene graphs for interactive object search with mobile manipulation. *IEEE Robotics and Automation Letters*, 2024.

Nathan Hughes, Yun Chang, and Luca Carlone. Hydra: A real-time spatial perception system for 3d scene graph construction and optimization. *arXiv preprint arXiv:2201.13360*, 2022.

Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Alaa Maalouf, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, et al. Conceptfusion: Open-set multimodal 3d mapping. *arXiv preprint arXiv:2302.07241*, 2023.

Yuanchen Ju, Yongyuan Liang, Yen-Jen Wang, Nandiraju Gireesh, Yuanliang Ju, Seungjae Lee, Qiao Gu, Elvis Hsieh, Furong Huang, and Koushil Sreenath. Momagraph: State-aware unified scene graphs with vision-language model for embodied task planning. *arXiv preprint arXiv:2512.16909*, 2025.

Sebastian Koch, Narunas Vaskevicius, Mirco Colosi, Pedro Hermosilla, and Timo Ropinski. Open3dsg: Open-vocabulary 3d scene graphs from point clouds with queryable objects and open-set relationships. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14183–14193, 2024.

Long Lian, Yifan Ding, Yunhao Ge, Sifei Liu, Hanzi Mao, Boyi Li, Marco Pavone, Ming-Yu Liu, Trevor Darrell, Adam Yala, et al. Describe anything: Detailed localized image and video captioning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 21766–21777, 2025.

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

Timo Lüddecke and Alexander Ecker. Image segmentation using text and image prompts. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 7086–7096, 2022.

Dominic Maggio and Luca Carlone. Bayesian fields: Task-driven open-set semantic gaussian splatting. *arXiv preprint arXiv:2503.05949*, 2025.

Dominic Maggio, Yun Chang, Nathan Hughes, Matthew Trang, Dan Griffith, Carlyn Dougherty, Eric Cristofalo, Lukas Schmid, and Luca Carlone. Clio: Real-time task-driven open-set 3d scene graphs. *IEEE Robotics and Automation Letters*, 2024.

Dominic Maggio, Nicolas Gorlo, and Luca Carlone. Found-it: Foundation-model-first task-driven 3d scene graphs with granularity on demand. *arXiv preprint arXiv:2605.25371*, 2026.

OpenAI. Introducing GPT-5.4. `https://openai.com/index/introducing-gpt-5-4/`, March 2026. Accessed: July 14, 2026.

Albert Gassol Puigjaner, Angelos Zacharia, and Kostas Alexis. Relationship-aware hierarchical 3d scene graph for task reasoning. *arXiv preprint arXiv:2602.02456*, 2026.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

Adam Rashid, Satvik Sharma, Chung Min Kim, Justin Kerr, Lawrence Yunliang Chen, Angjoo Kanazawa, and Ken Goldberg. Language embedded radiance fields for zero-shot task-oriented grasping. In *7th Annual Conference on Robot Learning*, 2023.

Antoni Rosinol, Andrew Violette, Marcus Abate, Nathan Hughes, Yun Chang, Jingnan Shi, Arjun Gupta, and Luca Carlone. Kimera: From slam to spatial perception with 3d dynamic scene graphs. *The International Journal of Robotics Research*, 40(12-14):1510–1546, 2021.

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

Dhruv Shah, Błażej Osiński, Sergey Levine, et al. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on robot learning*, pp. 492–504. PMLR, 2023.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on robot learning*, pp. 894–906. PMLR, 2022.

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

Shun-Cheng Wu, Johanna Wald, Keisuke Tateno, Nassir Navab, and Federico Tombari. Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7515–7525, 2021.

Ziyi Xia, Chaoran Xiong, Litao Wei, Xinhao Hu, and Ling Pei. Exploring bottlenecks in vlm-llm navigation: How 3d scene understanding capability impacts zero-shot vln. *arXiv preprint arXiv:2605.14801*, 2026.

Kashu Yamazaki, Taisei Hanyu, Khoa Vo, Thang Pham, Minh Tran, Gianfranco Doretto, Anh Nguyen, and Ngan Le. Open-fusion: Real-time open-vocabulary 3d mapping and queryable scene representation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9411–9417. IEEE, 2024.

Zhijie Yan, Shufei Li, Zuoxu Wang, Lixiu Wu, Han Wang, Jun Zhu, Lijiang Chen, and Jihong Liu. Dynamic open-vocabulary 3d scene graphs for long-term language-guided mobile manipulation. *IEEE Robotics and Automation Letters*, 2025.

Hang Yin, Xiuwei Xu, Zhenyu Wu, Jie Zhou, and Jiwen Lu. Sg-nav: Online 3d scene graph prompting for llm-based zero-shot object navigation. *Advances in neural information processing systems*, 37:5285–5307, 2024.

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

A Appendix

A.1 JITOMA-Bench Details

A.1.1 Dataset Statistics

JITOMA-Bench contains 126 evaluation task instances across the Cubicle, Office, and Apartment scenes. As summarized in Fig. 5, the three scenes contain 38, 42, and 46 task instances, respectively, spanning all three evaluation tiers. Across the benchmark, these tasks refer to 80 unique annotated 3D ground-truth boxes. Tier 2 intentionally reuses Tier 1 objects within continuous task streams; therefore, repeated task–object references across tiers should not be interpreted as additional unique ground-truth instances.

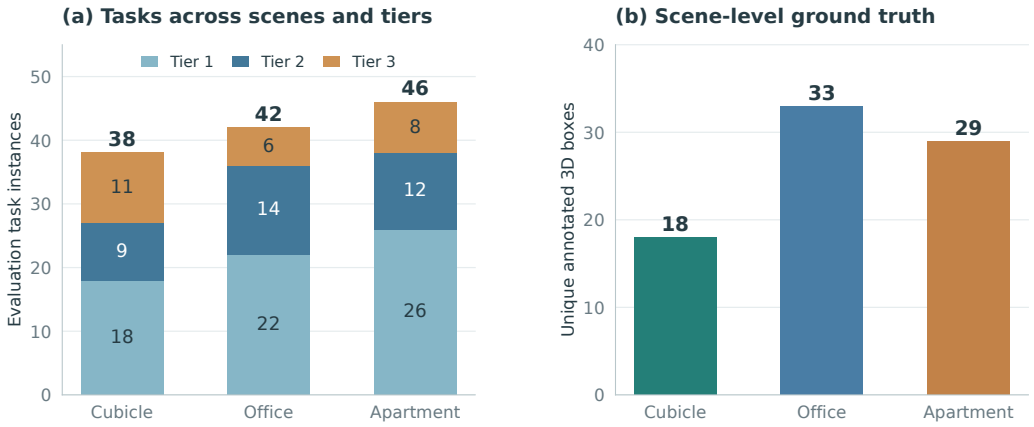

Figure 5: **Composition of JITOMA-Bench. (a)** Number of evaluation tasks in each scene, decomposed by Tier 1, Tier 2, and Tier 3. **(b)** Number of unique annotated 3D ground-truth boxes per scene. JITOMA-Bench contains 126 task instances and 80 unique GT boxes in total; Tier 2 reuses Tier 1 tasks and ground truth under a continual evaluation protocol.

The benchmark also covers a broad object vocabulary rather than repeatedly querying a small set of categories. Fig. 6 aggregates all task-target references across the three tiers, yielding 156 references in total: 66 from Tier 1, 35 from Tier 2, and 55 from Tier 3. Frequently referenced objects such as *backpack* and *hat* occur across multiple tasks and tiers, while most object labels appear only a small number of times. This long-tailed distribution reflects the open-vocabulary nature of the underlying scenes and reduces dependence on a small set of repeatedly queried categories. For visualization, selected surface-form aliases are merged, while the underlying ground-truth instances remain unchanged.

Tier 3 further increases compositional complexity by requiring each instruction to ground multiple objects. As shown in Fig. 7, the 25 Tier 3 tasks contain 25 primary and 30 latent target references. Twenty tasks involve two targets and five involve three, with all three scenes contributing multi-object instructions. The latent targets span diverse functional roles, including containers, destinations, tools, serving destinations, cleaning supplies, reference materials, and supporting objects. We retain the role descriptions assigned during annotation rather than post-hoc merging them into broader categories, preserving the functional diversity of the original instructions.

A.1.2 Task Annotation Protocol

Tier 1 provides the base set of explicit single-object grounding tasks. Each task is evaluated independently using the corresponding Clio ground-truth object annotations.

For Tier 2, we manually annotate the temporal visibility interval of each target along the original RGB-D trajectory. A task becomes active at its first annotated visible frame and is queried ten frames after its last-visible frame. Fig. 8 visualizes these intervals over recorded trajectory time. The Cubicle, Office, and Apartment sequences contain 9, 14, and 12 Tier 2 tasks over 91.1 s, 194.6 s, and

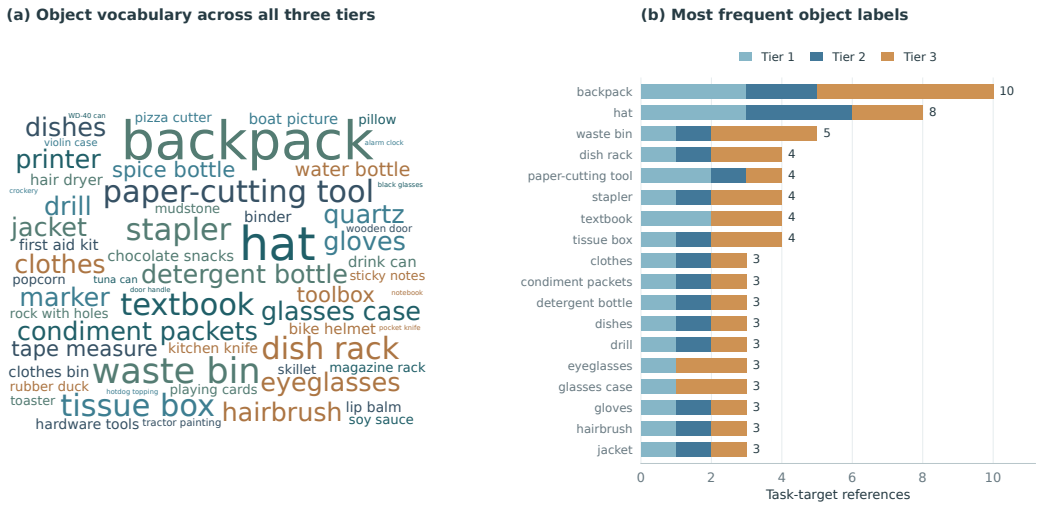

Figure 6: **Object vocabulary and target frequency in JITOMA-Bench.** **(a)** Object vocabulary aggregated across all three tiers. **(b)** Most frequently referenced object labels, decomposed by tier. The benchmark contains 156 task-target references in total (66/35/55 for Tier 1/2/3). Counts include intentional Tier 2 reuse; display aliases merge selected surface forms rather than ground-truth instances.

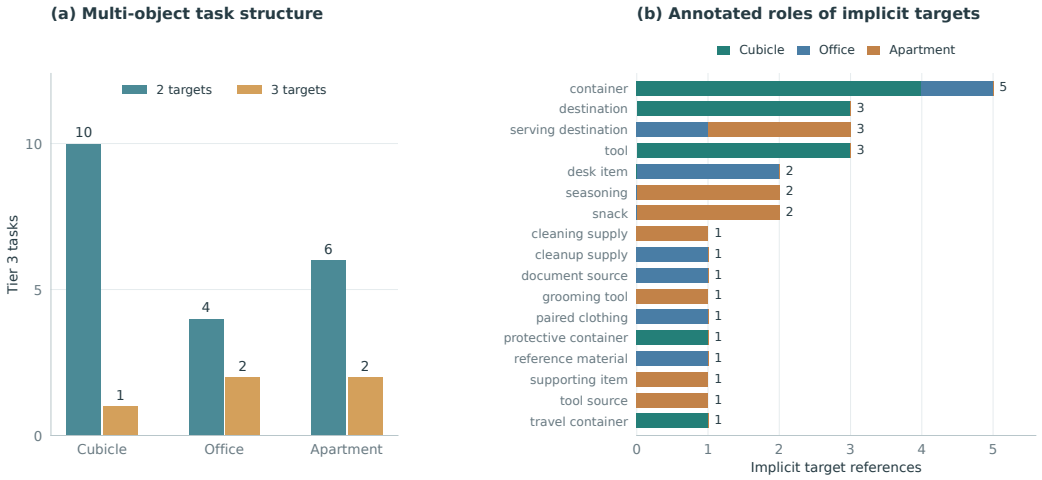

Figure 7: **Structure of Tier 3 complex tasks.** **(a)** Distribution of two- and three-target instructions across the three scenes. **(b)** Functional roles assigned to latent targets. Tier 3 contains 25 tasks with 25 primary and 30 latent target references. Role names are preserved directly from the annotations without post-hoc category merging.

155.7 s of recorded RGB-D data, respectively. These task windows are annotation intervals rather than inferred runtime concurrency; the ten-frame query offset therefore also does not correspond to a fixed delay in seconds. This protocol preserves a persistent scene memory throughout the complete trajectory and evaluates the system under repeated task transitions without resetting the map between queries.

Tier 3 is constructed from the same scene annotations but replaces atomic object queries with multi-object instructions containing both explicit primary objects and execution-implied latent objects. Candidate instructions are generated from Clio ground-truth labels and then manually assigned primary and latent targets. Three PhD researchers independently review each candidate, and an instruction is discarded when at least two reviewers identify ambiguity, infeasibility, or unnecessary latent

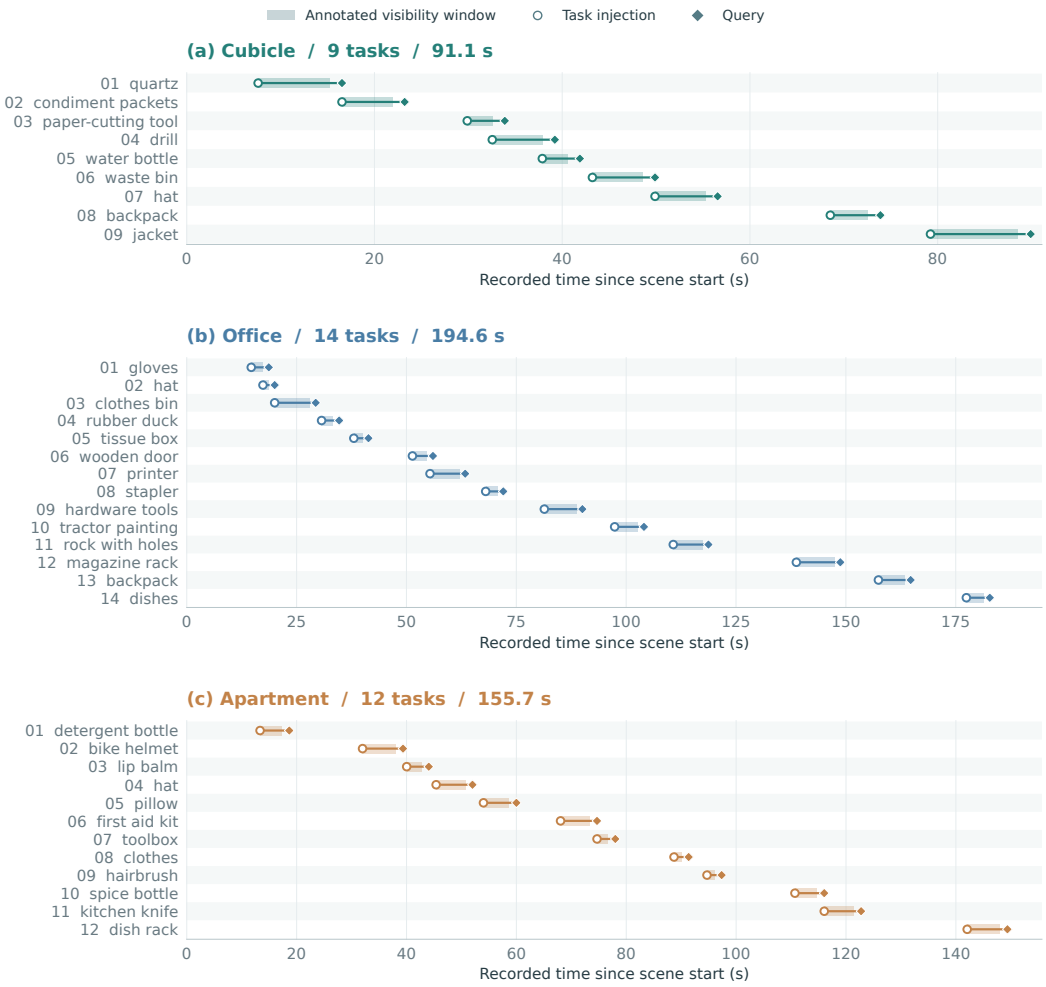

Figure 8: **Tier 2 temporal task annotations.** Annotated target-visibility intervals, task-injection times, and query times for the Cubicle, Office, and Apartment trajectories. Tasks are injected at the first-visible frame and evaluated ten frames after the annotated last-visible frame. Time is shown using the recorded RGB-D timestamps; the visibility windows are manual annotations and should not be interpreted as inferred runtime concurrency.

targets. The resulting 25 tasks contain the multi-object and functional-role distributions summarized in Fig. 7.

A.2 Controlled Study of Perceptual Saturation

A.2.1 Prompt and Graph Construction

To isolate the effect of task-irrelevant semantic context, the AOT and JIT conditions use exactly the same task instruction, planner, prompt template, output format, and decoding configuration. The only difference is the set of scene-graph entities exposed to the planner. All task-relevant entities retain identical identifiers, object names, and geometric attributes across conditions. The JIT condition contains only the entities required for task execution, whereas the AOT condition additionally includes task-irrelevant objects from the same scene.

For the qualitative example in Sec. 3, the task is "pack my eyeglasses for travel" in the Cubicle scene. The task-relevant condition contains the three required entities (*eyeglasses*, *glasses case*,

and *backpacks*), while the full-graph condition contains the same three entries together with 15 additional distractors.

AOT full graph.

Ahead-of-Time (AOT) Full Graph

Task: pack my eyeglasses for travel
Scene: cubicle

Scene graph:

```
- id="full_001", object="condiment packets"
↪ coords=[center=(-0.83, 1.62, -0.76), extents=(0.27, 0.19, 0.01)]
- id="full_002", object="drink cans"
↪ coords=[center=(-0.41, 4.03, -0.83), extents=(0.14, 0.15, 0.09)]
- id="full_003", object="eyeglasses"
↪ coords=[center=(-0.83, 1.26, -0.70), extents=(0.10, 0.03, 0.02)]
- id="full_004", object="glasses case"
↪ coords=[center=(-0.82, 1.27, -0.72), extents=(0.14, 0.04, 0.02)]
- id="full_005", object="grey jacket"
↪ coords=[center=(-0.02, 0.72, -0.71), extents=(0.72, 0.26, 0.15)]
- id="full_006", object="my silver water bottle"
↪ coords=[center=(-0.52, 4.01, -0.77), extents=(0.23, 0.05, 0.04)]
- id="full_007", object="notebooks"
↪ coords=[center=(-0.26, 3.70, -0.86), extents=(0.66, 0.34, 0.03)]
- id="full_008", object="mudstone rock"
↪ coords=[center=(-1.00, 0.11, -0.63), extents=(0.13, 0.07, 0.06)]
- id="full_009", object="tool to cut paper"
↪ coords=[center=(-0.77, 2.86, -0.81), extents=(0.30, 0.13, 0.03)]
- id="full_010", object="sticky notes"
↪ coords=[center=(-1.22, 1.32, -0.65), extents=(0.18, 0.18, 0.01)]
- id="full_011", object="textbooks"
↪ coords=[center=(-1.04, 2.08, -0.73), extents=(0.60, 0.41, 0.08)]
- id="full_012", object="waste bins"
↪ coords=[center=(0.17, 3.27, -1.36), extents=(0.83, 0.55, 0.26)]
- id="full_013", object="hats"
↪ coords=[center=(0.31, 3.57, -0.82), extents=(0.34, 0.22, 0.08)]
- id="full_014", object="backpacks"
↪ coords=[center=(0.75, 2.56, -0.82), extents=(0.61, 0.59, 0.21)]
- id="full_015", object="red crockery"
↪ coords=[center=(-0.87, 3.92, -0.84), extents=(0.47, 0.27, 0.09)]
- id="full_016", object="hardware drill"
↪ coords=[center=(-0.86, 3.58, -0.83), extents=(0.20, 0.17, 0.04)]
- id="full_017", object="quartz rock"
↪ coords=[center=(-0.91, 0.09, -0.64), extents=(0.10, 0.08, 0.04)]
- id="full_018", object="tape measure"
↪ coords=[center=(-0.72, 3.61, -0.83), extents=(0.09, 0.08, 0.06)]
```

JIT task-relevant graph.

Just-In-Time (JIT) Task-Relevant Graph

Task: pack my eyeglasses for travel
Scene: cubicle

Scene graph:

```
- id="full_003", object="eyeglasses"
↪ coords=[center=(-0.83, 1.26, -0.70), extents=(0.10, 0.03, 0.02)]
- id="full_004", object="glasses case"
↪ coords=[center=(-0.82, 1.27, -0.72), extents=(0.14, 0.04, 0.02)]
- id="full_014", object="backpacks"
↪ coords=[center=(0.75, 2.56, -0.82), extents=(0.61, 0.59, 0.21)]
```

Shared planner prompt. The following prompt is used unchanged for both graph conditions:

Shared Task-Planning Prompt

Input Prompt:
Return the answer in exactly this style, with 3 or 4 numbered quoted steps and nothing else:

1. `"Action using object [center=(x, y, z)]",`
2. `"Next action using object [center=(x, y, z)]",`
3. `"Next action using object [center=(x, y, z)]",`
4. `"Final action using object [center=(x, y, z)]"`

Rules:

1. Mention all objects that are required to complete the task.
2. If a destination, container, support, source, or tool is needed and exists in the graph, include it.
3. Every mentioned object must include its center coordinate exactly once in that step.
4. Use short imperative actions. Do not explain.

A.2.2 Controlled Distractor Analysis

To examine whether the qualitative failure mode in Sec. 3 extends beyond a single example, we conduct a controlled distractor study over the Tier 3 tasks of JITOMA-Bench. For each task, all task-required entities and their attributes are held fixed, while $k \in \{0, 5, 10, 15\}$ task-irrelevant entities from the same scene are added to the planner input. All conditions use the same planner, prompt template, and output format described in Appendix A.2.1; thus, the manipulated factor is the amount of irrelevant semantic context presented alongside the task-relevant graph. Across the three scenes and repeated distractor configurations, the study contains 1,194 planner requests, all of which complete without interface errors or output truncation.

Evaluation metric. We measure *target coverage*, defined as the fraction of annotated task-required objects explicitly referenced by the generated plan. We report coverage over all required targets and separately over *latent targets*, i.e., objects not explicitly named in the command but required for task execution. The latter provides a more demanding measure of whether the planner recovers implicit functional dependencies such as containers, destinations, or tools.

Table 3: Controlled distractor analysis on Tier 3 of JITOMA-Bench. Task-relevant graph content is held fixed while k task-irrelevant entities are added. Results are averaged across the three scenes.

Distractors k	All-target coverage (%)	Latent-target coverage (%)
0	82.31	68.10
5	79.26	62.08
10	78.23	60.97
15	75.24	56.30
$\Delta(15-0)$	-7.07	-11.81

Effect of irrelevant semantic load. As shown in Table 3, target coverage decreases as additional task-irrelevant entities are introduced. From $k = 0$ to $k = 15$, all-target coverage drops from 82.31% to 75.24%, while latent-target coverage decreases more substantially from 68.10% to 56.30%. A paired bootstrap comparison between these two conditions yields 95% confidence intervals of $[-13.89, -1.42]$ percentage points for all-target coverage and $[-23.21, -2.12]$ for latent-target coverage. The larger degradation on latent targets indicates that irrelevant semantic context particularly interferes with recovering objects implied by the execution chain rather than explicitly stated in the command.

This behavior is not explained by loss of the primary task target: primary-target coverage remains nearly unchanged, from 98.52% at $k = 0$ to 98.11% at $k = 15$. The latent-target degradation is also observed across all three scenes: coverage decreases from 100.00% to 93.33% in Cubicle,

67.22% to 47.22% in Office, and 37.08% to 28.33% in Apartment. Qualitative inspection reveals recurring failures such as omitting a required container or destination, terminating a multi-step plan prematurely, or introducing operations on irrelevant objects.

Interpretation. The effect is not deterministic for every individual task. Between $k = 0$ and $k = 15$, latent-target coverage decreases for 9 of the 25 Tier 3 tasks, remains unchanged for 14, and improves for 2. We therefore interpret perceptual saturation as a population-level tendency under increasing irrelevant semantic load rather than a universal failure mode. Together with the qualitative example in Sec. 3, these results provide controlled evidence that exposing a planner to additional task-irrelevant scene semantics can reduce recovery of task-required information, with the strongest effect on latent functional dependencies.

A.3 Deployment Details

A.3.1 Hardware and Software Environment

Each experimental run uses a single NVIDIA RTX A6000 GPU with 48 GB VRAM and is executed on Ubuntu 22.04. The main JITOMA pipeline and the local Qwen service share the same GPU but run in separate software environments. The main pipeline uses Python 3.10, PyTorch 2.0.1 with CUDA 11.8, and Transformers 4.41.2, while the local LLM service uses Python 3.10, PyTorch 2.4.0 with CUDA 12.1, and Transformers 5.6.0.dev0. Qwen3.5-9B is served locally in BF16 precision.

Runtime measurement. TPF is computed as the total wall-clock time required to process a complete scene replay divided by the number of processed RGB-D frames. The measurement covers the full online mapping pipeline, including perception, memory updates, task-conditioned processing, and on-demand semantic operations such as DAM captioning whenever they are triggered. The same timing protocol is applied to all methods.

A.3.2 Hyperparameter Settings

JITOMA contains a small set of hyperparameters governing three stages of the just-in-time growth process: task-conditioned observation admission, memory lifecycle management, and on-demand semantic activation. Table 4 summarizes the default values used in all experiments.

Table 4: Default hyperparameters of JITOMA.

Hyperparameter	Symbol	Value
Observation admission ratio	r	0.5
Heatmap pooling ratio	ρ	0.2
Anchor promotion threshold	N_{prom}	8 observations
Ephemeral expiration threshold	T_{eph}	20 frames
Activation candidates per concept	K_{act}	3
Heatmap evidence weight	λ	0.10

Task-guided observation admission. During active tasks, JITOMA ranks tracked observations according to their alignment with the task-conditioned heatmap. Let H_t^q denote the 2D heatmap generated for command q at frame t. For each tracked observation o, we compute an admission score

$$s_t(o, q) = 0.3\,\overline{H}^q_{t,\text{mask}}(o) + 0.7\,\overline{H}^q_{t,\text{top-}\rho}(o), \tag{6}$$

where the first term represents the average heatmap response inside the object mask, and the second term captures localized high-response regions within the top ρ fraction of pixels. We set $\rho = 0.2$ in all experiments.

Given N_t eligible tracked observations at frame t, JITOMA retains the top

$$K_{o,t} = \lceil rN_t \rceil \tag{7}$$

observations according to $s_t(o, q)$, where $K_{o,t}$ denotes the number of observations admitted into memory and $r = 0.5$ is the admission ratio. This adaptive budget allows the frontend to allocate memory updates according to the current observation density rather than using a fixed number of admitted observations.

Task-free conservative writing When no task query is active, JITOMA does not apply task-conditioned heatmap gating. Instead, it maintains a lightweight task-agnostic writing policy whose goal is to establish stable object evidence while avoiding unnecessary repeated memory updates. Newly observed tracks are admitted when they satisfy the tracking confidence criterion and have not previously contributed observations to memory. For previously written tracks, JITOMA only performs additional updates when they provide sufficiently improved visual evidence. This distinction prevents persistent memory from being dominated by repeated observations of the same object while still allowing newly emerging or higher-quality views to refine the stored representation.

For repeated observations, JITOMA evaluates whether a new view provides meaningful visual improvement using an image-quality proxy. For an observation $o_{t,i}$, the quality score is defined as

$$Q_{t,i} = 0.55A_{t,i} + 0.15B_{t,i} + 0.30P_{t,i}, \tag{8}$$

where $A_{t,i}$, $B_{t,i}$, and $P_{t,i}$ represent normalized object area, image centeredness, and tracking confidence, respectively. The score provides a lightweight measure of observation quality without requiring semantic understanding or additional 3D reconstruction.

Let $e_i(t)$ denote the number of observations previously written for track i, and let $Q^{\star}_{t-1,i}$ denote its best historical quality score. A previously written track is updated when

$$e_i(t) < 3 \vee \left(Q^{\star}_{t-1,i} < 0.80 \wedge Q_{t,i} - Q^{\star}_{t-1,i} \geq 0.10\right). \tag{9}$$

The first three writes allow initial evidence accumulation. After this stage, additional updates are accepted only when the observation provides a sufficient quality improvement. To avoid excessive updates across many tracks, at most three repeated tracks are selected per frame according to update priority.

Memory lifecycle. Admitted observations are first maintained as ephemeral hypotheses before being promoted to persistent dormant anchors. A hypothesis is promoted when its associated track accumulates $N_{\text{prom}} = 8$ observations, using temporal observation support as a lightweight persistence criterion rather than performing semantic verification at this stage.

An unpromoted hypothesis is removed after $T_{\text{eph}} = 20$ inactive frames, where the threshold is measured from the latest observation update. This expiration rule only applies to ephemeral states; promoted anchors remain in memory for future task-conditioned activation. The global memory capacity is fixed to 512 nodes, including both ephemeral hypotheses and dormant anchors. Importantly, dormant storage does not imply semantic expansion: captions and functional structures are generated only after an anchor is activated by a task query.

On-demand semantic activation. When a task query arrives, JITOMA retrieves relevant dormant anchors before performing expensive semantic expansion. For each concept $c \in C(q)$, at most $K_{\text{act}} = 3$ candidate anchors are retrieved from V^{anc} using the CLIP visual key and task-conditioned evidence:

$$S(v_j, c) = \cos(\text{CLIP}_{\text{text}}(c), z_j) + \lambda \tilde{h}^c_j. \tag{10}$$

where z_j denotes the stored visual key of anchor v_j and $\tilde{h}^c_j$ is the normalized accumulated heatmap evidence for concept c. We set $\lambda = 0.10$, providing a small perceptual correction while keeping visual-semantic similarity as the dominant retrieval signal. Only retrieved candidates undergo captioning and subsequent semantic expansion; all remaining anchors stay dormant.

A.3.3 HYPERPARAMETER SENSITIVITY ANALYSIS

We investigate the sensitivity of JITOMA to three hyperparameters that directly control information admission and semantic expansion: the observation admission ratio r, the activation candidate budget K_{act}, and the heatmap evidence weight λ. We perform one-factor-at-a-time experiments on Tier 2 of JITOMA-Bench, while fixing all remaining parameters to the default configuration $(r, K_{\text{act}}, \lambda) = (0.5, 3, 0.1)$. Each configuration is evaluated on the three benchmark scenes with identical task parser outputs and the same tracking, memory, and geometry processing settings.

We report mIoU, mR@1, mR@3, and TPF following the main evaluation protocol. To measure the extent of semantic expansion, we additionally report Accumulated Objects (Acc. Obj.), which counts the number of unique object nodes that receive semantic captions throughout the replay. Unlike Peak, which measures the maximum instantaneous active semantic-state size, Acc. Obj. captures cumulative semantic materialization over time.

Table 5: Hyperparameter sensitivity analysis of JITOMA on Tier 2 of JITOMA-Bench. Only the parameter indicated in each block is varied while all other parameters remain fixed to the default configuration. Results are averaged across the three scenes. Default settings are marked with $\star$.

Setting	mIoU ↑	mR@1 ↑	mR@3 ↑	TPF ↓	Acc. Obj. ↓
Observation admission ratio ($K_{\text{act}} = 3$, $\lambda = 0.1$)					
$r = 0.25$	23.35	54.45	**67.24**	0.2781	82.67
$r = 0.50^{\star}$	**23.95**	**56.83**	67.11	0.2794	84.33
$r = 0.75$	23.40	55.11	63.36	0.2790	84.00
$r = 1.00$	23.18	54.76	64.59	0.2757	83.00
Activation candidate budget ($r = 0.5$, $\lambda = 0.1$)					
$K_{\text{act}} = 1$	22.53	51.19	51.19	0.2624	34.67
$K_{\text{act}} = 2$	23.83	57.14	65.87	0.2704	63.00
$K_{\text{act}} = 3^{\star}$	23.95	56.83	67.11	0.2794	84.33
$K_{\text{act}} = 5$	23.81	53.75	65.43	0.2952	109.33
$K_{\text{act}} = 8$	**24.81**	**58.29**	**68.83**	0.3124	147.00
Heatmap evidence weight ($r = 0.5$, $K_{\text{act}} = 3$)					
$\lambda = 0$	21.49	51.90	64.02	0.2742	58.33
$\lambda = 0.05$	21.09	48.32	63.10	0.2742	68.33
$\lambda = 0.10^{\star}$	23.95	**56.83**	**67.11**	0.2794	84.33
$\lambda = 0.20$	**24.02**	56.04	66.31	0.2820	89.00

Observation admission ratio. The observation admission ratio controls how much task-conditioned visual evidence is committed by the frontend. The default setting $r = 0.5$ achieves the strongest overall grounding performance, obtaining the highest mIoU (23.95) and mR@1 (56.83) among all tested ratios. Reducing the ratio to 0.25 slightly improves mR@3, but decreases mIoU and mR@1, suggesting that overly aggressive filtering may remove useful observations. Increasing the admission ratio beyond 0.5 does not consistently improve accuracy, indicating that admitting additional observations alone is insufficient for better grounding.

The relatively stable Acc. Obj. and TPF across different values of r suggest that this parameter mainly influences the relevance of admitted evidence rather than the total semantic expansion cost. We therefore retain $r = 0.5$ as a balanced operating point between preserving task-related observations and limiting unnecessary memory updates.

Activation candidate budget. The activation candidate budget K_{act} controls the amount of semantic expansion performed after task activation and introduces the clearest accuracy–efficiency trade-off. A larger candidate budget provides more opportunities to recover relevant anchors, but also increases the number of objects exposed to semantic processing.

Increasing K_{act} from 3 to 8 improves mR@1 and mR@3 from 56.83/67.11 to 58.29/68.83. However, this improvement increases Acc. Obj. from 84.33 to 147.00 and raises TPF from 0.2794s to 0.3124s. Conversely, smaller budgets reduce semantic expansion but provide fewer candidate anchors for retrieval. Although $K_{\text{act}} = 8$ achieves slightly higher accuracy, we select $K_{\text{act}} = 3$ as the

default configuration because it provides strong grounding performance while maintaining bounded on-demand semantic growth.

Heatmap evidence weight. The heatmap evidence weight λ determines the contribution of task-conditioned perceptual evidence during anchor ranking. Removing this term ($\lambda = 0$) decreases mIoU, mR@1, and mR@3 by 2.46, 4.94, and 3.09 percentage points compared with the default setting, respectively. This demonstrates that task-conditioned evidence complements CLIP-based visual similarity during anchor activation.

Increasing λ to 0.2 provides only marginal accuracy improvement while slightly increasing semantic expansion. The results indicate that task evidence is most effective as a complementary signal rather than a replacement for visual-semantic matching. We therefore retain $\lambda = 0.1$ as a moderate weighting factor.

Overall, these sensitivity experiments support the default configuration $(r, K_{\mathrm{act}}, \lambda) = (0.5, 3, 0.1)$ as a balanced operating point, providing strong grounding accuracy while preserving the bounded semantic growth behavior targeted by JITOMA.