ProCompNav: Proactive Instance Navigation with Comparative Judgment for Ambiguous User Queries

Junhyuk Kwon[1], Seungjoon Lee[2], Hyejin Park[1], Kyle Min[3], Jungseul Ok[1,2]†
[1]GSAI, POSTECH [2]CSE, POSTECH [3]Oracle
{treejhk, sjlee1218, parkebbi2, jungseul}@postech.ac.kr
kyle.min@oracle.com
https://tree-jhk.github.io/procompnav/

Abstract

In human-robot interaction, agentic robots should resolve ambiguous user requests through proactive clarification while keeping user burden low. We study this challenge in interactive instance navigation, where an agent must identify and reach the user-intended target among similar objects. Existing approaches ask open-ended questions and select a candidate that strongly matches the accumulated user-provided information. However, this process can require lengthy user responses and lead the agent to make premature decisions without distinguishing the target from distractors. To address these limitations, we propose Proactive Instance Navigation with Comparative Judgment (ProCompNav), a two-stage framework that first constructs a candidate pool and then identifies the target through Recursive Comparative Judgment (RCJ). RCJ iteratively narrows the pool by selecting an attribute-value pair that divides the candidates, asking the user a binary question, and removing inconsistent candidates, without requiring an attribute unique to the target. ProCompNav substantially improves Success Rate and reduces Response Length on an interactive navigation benchmark, and achieves the highest Success Rate among evaluated methods on a non-interactive benchmark. In two human studies, participants preferred ProCompNav's binary questions to open-ended questions and its batched delivery to interleaved interaction. Our results highlight the need to jointly consider task performance and user-friendly interaction strategies when designing agentic robots.

1 Introduction

LLM-based agents support human-agent collaboration on digital tasks by clarifying user intent when requests are incomplete or ambiguous [20, 24, 27]. This challenge also arises in agentic robotics, where human requests may leave task goals or preferences unspecified, requiring robots to ask follow-up questions as they plan and act [9, 19]. In human-robot interaction, clarification should help robots accomplish the intended task while limiting user effort. For example, "Find the cabinet" gives a robot an object category to search for, but several cabinets may fit the request even when the user has a particular one in mind. What needs to be clarified can then depend on the objects the robot encounters. We focus on interactive instance navigation [22], aiming to identify and reach the intended instance through proactive clarification while keeping user burden low.

One approach to identifying the intended instance is *independent matching*, in which the agent asks the user open-ended questions about attributes of each detected candidate (e.g., color) and scores candidates based on the accumulated responses (e.g., blue) [22]. However, this can lead to premature selection of a distractor sharing many target attributes (Figure 1(a)). A natural way to mitigate this

†Corresponding author.

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

Table 1: Comparison of three disambiguation strategies on CoIN-Bench. We report Success Rate (SR), average total Response Length (RL) per episode, and average Number of Questions (NQ) over episodes with user interaction.

Decision Strategy	Stage		Val Seen			Val Seen Synonyms			Val Unseen		
	Pool	Compare	SR ↑	RL ↓	NQ ↓	SR ↑	RL ↓	NQ ↓	SR ↑	RL ↓	NQ ↓
(a) Independent Matching [22]	✗	✗	10.5	109.5	1.2	15.3	129.2	1.2	8.9	122.8	1.3
(b) Pooled Independent Matching	✓	✗	17.5	460.2	4.1	22.0	519.2	4.3	13.3	467.7	4.0
(c) Comparative Judgment (Ours)	✓	✓	**23.7**	**4.2**	2.2	**28.1**	**4.3**	2.2	**17.0**	**4.2**	2.3

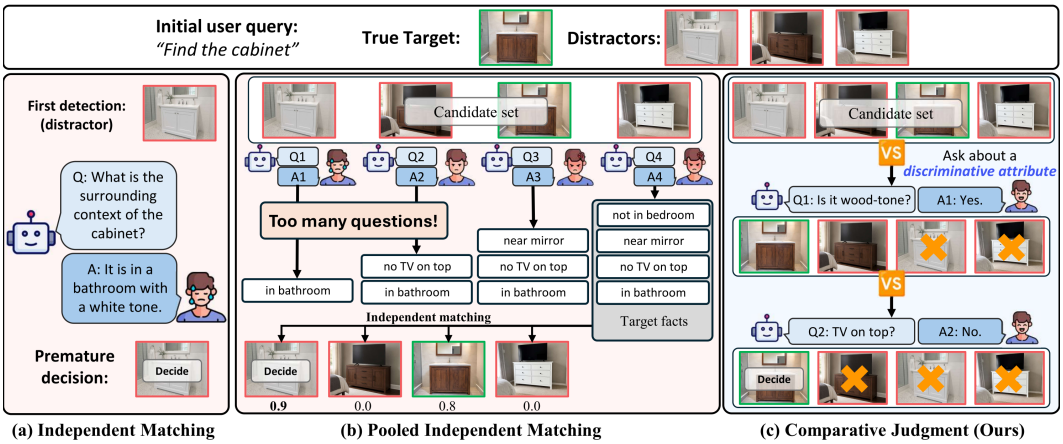


Figure 1: Three strategies for instance navigation under an ambiguous user query. (a) *Independent Matching* [22] evaluates each candidate independently. (b) *Pooled Independent Matching* collects multiple candidates, accumulates user responses about each candidate, and applies independent matching. (c) *Comparative Judgment* (Ours) constructs a candidate pool and asks discriminative binary questions based on candidate contrasts.

is *pooled independent matching*, which collects multiple candidates before making a decision and applies independent matching to each candidate. By accumulating more information, it increases the chance of distinguishing the target from distractors. However, the accumulated information may still be insufficient to distinguish similar distractors, while asking about every candidate substantially increases both the number of questions and the length of user responses (Figure 1(b)).

We propose Proactive Instance Navigation with Comparative Judgment (ProCompNav), an interactive instance navigation framework for identifying the user-intended target with robustness to distractors and low user burden. ProCompNav first constructs a candidate pool, and then disambiguates the target from the pool through Recursive Comparative Judgment (RCJ) (Figure 1(c)). RCJ iteratively narrows the pool by (i) selecting an attribute (e.g., material) and its value (e.g., metal) that divide the remaining candidates into two groups, (ii) asking the user whether the target has that value, and then (iii) removing candidates inconsistent with the answer. ProCompNav improves robustness to distractors by asking questions on attribute-value pairs that distinguish candidates even when they share many other attributes. ProCompNav also reduces user burden in two ways. First, each discriminative question can eliminate multiple candidates with a single answer, reducing the number of questions needed. Second, ProCompNav asks its questions consecutively after collecting candidates, reducing repeated, disruptive interruptions that draw users' attention back to the agent [2, 6].

We summarize our contributions as follows: (1) We propose ProCompNav, an interactive instance navigation framework that identifies the user-intended target despite distractors while reducing user burden. We achieve this by our core component, RCJ, which compares candidates, makes discriminative binary questions, and removes candidates inconsistent with user answers. (2) We evaluate ProCompNav on navigation benchmarks and in two human studies. On interactive CoIN-Bench, ProCompNav improves success rate while reducing response length. It also achieves the highest success rate among evaluated methods on the non-interactive TextNav benchmark. In our human studies, participants preferred ProCompNav's binary questions to open-ended questions and its batched delivery to interleaved interaction (Section R). ProCompNav shows how agentic robots can use their own observations to guide dialogue, helping people clarify their intent with brief answers.

2 Related Works

2.1 Language-driven instance navigation

Unlike the widely studied object navigation task [3, 4, 7, 14, 25, 26, 30-40], where reaching any instance of the target category is sufficient, language-driven instance navigation requires identifying a specific target instance among same-category distractors in a 3D environment. Training-based methods learn navigation policies from observations and descriptions [21], but often require substantial training compute and generalize poorly to unseen environments. Training-free methods have therefore emerged as alternatives [10, 29, 32, 41]. Most existing methods match observed candidates to a fixed target description rather than explicitly comparing the current candidates, leaving them vulnerable to distractors with similar attributes [10, 32, 41]. Specifically, UniGoal [32] matches goal and scene graphs using predefined node, edge, and topology scores, while Context-Nav [10] verifies candidates using a predefined set of spatial predicates. Such fixed matching and verification rules do not adapt to the discriminative attributes that distinguish the current candidates. 3D-Mem [29] incrementally builds a visual memory of explored regions and uses a VLM to guide navigation and select the target, but does not explicitly extract discriminative attributes from candidate contrasts, potentially leaving the target ambiguous among similar distractors. Moreover, most existing methods assume a detailed target description is provided upfront, limiting their applicability to ambiguous queries that require user clarification.

2.2 Interactive target disambiguation

As user requests are often ambiguous, a growing body of work studies how embodied agents can resolve task ambiguity through user clarification [13, 19]. However, existing object-disambiguation methods address candidates visible in a scene or found within a single room, rather than searching an unknown environment to discover them [16, 28]. Similarly, AmbResVLM [5] resolves ambiguous task descriptions through user clarification in a visible scene. Unlike these methods, AIUTA [22] explores unknown environments while disambiguating the target through user dialogue. However, AIUTA independently evaluates each encountered candidate against accumulated user responses, potentially mistaking a distractor sharing many target attributes for the true target. Its open-ended questions also require users to provide descriptive responses. In contrast, ProCompNav collects candidates before disambiguation and selects discriminative attributes from candidate contrasts, enabling binary questions that distinguish similar distractors with short user responses.

3 Problem Formulation

We study an interactive instance navigation task in an unknown 3D environment, where a robot starts from an arbitrary location and must identify the user-intended target instance T^*. The robot navigates using RGB-D observations with a restricted 30° field of view (FOV). At the beginning of each episode, the robot receives minimal natural language input: an open-vocabulary category-c (e.g., "cabinet"). Let $\mathcal{O}_c$ denote the set of all instances of category-c in the environment (unknown to the robot). The target is unique, $T^* \in \mathcal{O}_c$, and the remaining instances are distractors $\mathcal{D} = \mathcal{O}_c \setminus \{T^*\}$. Since the category-$c$ may not uniquely specify an instance, T^* can be indistinguishable from the distractors $\mathcal{D}$ with minimal information. To resolve such ambiguities, the robot can engage in a natural language dialogue with the user during navigation, without exchanging any visual information [22]. At each timestep, the robot can take four discrete actions: `move_forward`, `turn_left`, `turn_right`, and `stop`, and may optionally ask a free-form natural-language question. An episode terminates when the robot executes `stop` or the predefined episode horizon is reached. Success is declared if the robot executes `stop` within distance r of T^*, within the horizon.

Because distractors $\mathcal{D}$ and T^* share many attribute-value pairs, (pooled) independent matching [22] is reliable only when the available target information suffices to distinguish T^* from $\mathcal{D}$. We instead identify T^* through *comparative judgment*: the robot accumulates a candidate pool of category-c instances and, at each round, prunes it with an attribute-value pair that splits the pool into two non-empty groups, until only T^* remains.

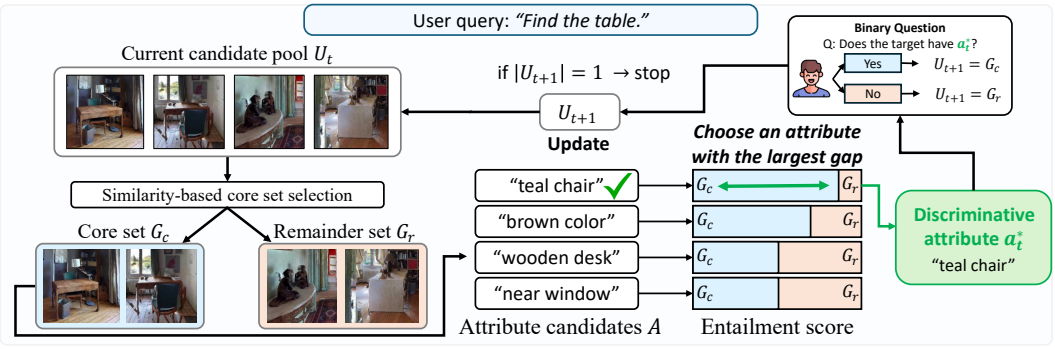


Figure 2: **Recursive Comparative Judgment.** At iteration t, ProCompNav splits the candidate pool U_t into a core set G_c and a remainder set G_r by similarity. It identifies a discriminative attribute a_t^*, that is common in G_c but not in G_r. Finally, it asks whether the target has a_t^*, and prunes the pool to obtain the next candidate pool U_{t+1} based on the user's response.

4 Proposed Method

4.1 Overview

We propose Proactive Instance Navigation with Comparative Judgment (**ProCompNav**), which first constructs a candidate pool and then compares candidates to ask discriminative questions that prune the pool. In the Pool Construction Stage (Sec 4.2), ProCompNav explores the environment to construct a candidate pool of category-c instances. In the Recursive Comparison Stage (Sec 4.3), ProCompNav replaces independent matching with comparative judgment across candidates, and minimizes user burden by asking only binary yes/no questions that can eliminate multiple candidates.

4.2 Pool construction stage

The Pool Construction Stage explores the unknown environment to collect the candidate pool $\mathcal{C} = \{c_i\}_{i=1}^{N}$ of N distinct candidates on which the Recursive Comparison Stage operates. Each candidate $c_i = (I_i, d_i)$ is represented by a multi-view collage image I_i and a multi-view description d_i. For each candidate i, the agent maintains an accumulated 3D point cloud $\mathcal{P}_i$ of the instance and a set of RGB views $\mathcal{V}_i$ captured from different viewpoints during exploration. For each new detection of category-c, we extract a 3D point cloud from the detected region. If it sufficiently overlaps with the accumulated point cloud $\mathcal{P}_i$ of an existing candidate i, the detection is assigned to candidate i and its RGB view is added to $\mathcal{V}_i$. Otherwise, a new candidate is initialized with its point cloud and view set seeded from this detection. From $\mathcal{V}_i$, we cluster the views in an embedding space into K clusters and select a representative per cluster, arrange the K representatives into the multi-view collage I_i, and prompt an MLLM to produce the multi-view description d_i. The description summarizes the candidate's attribute-value pairs—attributes (e.g., color, nearby objects) and their values (e.g., blue, next to a TV)—aggregated across the K views, capturing pairs that may be missed from any single viewpoint. Once $|\mathcal{C}| \geq N_{\min}$, the Pool Construction Stage terminates and the agent transitions to the Recursive Comparison Stage. Implementation details and example multi-view candidates are provided in Appendix B.

4.3 Recursive comparison stage

We propose **Recursive Comparative Judgment** (RCJ, Fig. 2), which identifies the target T^* by iteratively pruning the candidate pool $\mathcal{C}$ through binary questions on attribute-value pairs that contrast candidates. At each interaction round, ProCompNav extracts an attribute-value pair—a *Discriminative Attribute (DA)* that one group of candidates in the current pool has but the rest lack—asks the user whether T^* has it, and removes the inconsistent group. Crucially, RCJ does not require an attribute that uniquely identifies T^* at any round. Each round only needs an attribute-value pair that splits the current pool into two non-empty groups. Formally, let $U_t \subseteq \mathcal{C}$ denote the active candidate set at interaction round t, initialized as $U_0 = \mathcal{C}$. At iteration t, ProCompNav discovers a DA a_t^* that partitions U_t. After asking the user whether T^* possesses a_t^* and receiving their binary answer,

ProCompNav updates the active candidate set to U_{t+1} by retaining only the candidates consistent with this answer. This recursive pruning continues until $|U_t| = 1$. We next describe how core selection helps find an attribute that partitions U_t.

4.3.1 Similarity-based core set selection

When $|U_t| > 2$, we select a similar pair as the core set G_c and place the other candidates in the remainder set $G_r = U_t \setminus G_c$. In our ablation, this core-remainder split yields higher SR than KMeans grouping (Table 5, Appendix I). We measure the cohesion of a subset $V \subseteq U_t$ by its mean pairwise similarity:

$$\rho(V) = \frac{1}{|V|(|V|-1)} \sum_{i,j \in V, i \neq j} S(i,j) \ ,$$

where $S(i,j)$ is the pairwise similarity between instances i and j (defined below).

We use greedy peeling: starting from $V_0 = U_t$, we repeatedly remove the candidate with the lowest total similarity to the others until two candidates remain:

$$V_{l+1} = V_l \setminus \{i'\}, \quad \text{where } i' = \arg\min_{i \in V_l} \sum_{j \in V_l, j \neq i} S(i,j) \ .$$

We take the surviving pair as the core set, $G_c = V_{|U_t|-2}$, and set $G_r = U_t \setminus G_c$. Removing the least similar candidate makes ρ non-decreasing along the peeling path, so the surviving pair has the highest ρ on that path. When $|U_t| = 2$, we skip peeling and treat each candidate as a singleton group. We select the attribute with the largest entailment-score difference between them, and the binary answer updates the active set as described below.

We compute the pairwise similarity $S(i,j)$ by averaging text and visual similarities:

$$S(i,j) = \tfrac{1}{2} \left(\langle \hat{e}_i^{\text{text}}, \hat{e}_j^{\text{text}} \rangle + \left\langle \hat{e}_i^{\text{img}}, \hat{e}_j^{\text{img}} \right\rangle \right) \ ,$$

where $\hat{e}^{\text{text}}$ and $\hat{e}^{\text{img}}$ denote the ℓ_2-normalized textual and visual embeddings extracted from the instance's description and image, respectively.

4.3.2 Discriminative attribute (DA) discovery

To identify the target T^*, we aim to discover a DA a_t^* that contrasts the core set G_c against the remainder set G_r and enables effective pruning with a yes/no question. This requires judging whether each instance description supports a candidate attribute. We use a Natural Language Inference (NLI) model to verify them: given the description d_i of an instance i and a hypothesis of the form "instance i has attribute a", the NLI model provides a standardized judgment (entails, contradicts, or neutral) that yields stable scores for quantifying group-level contrast.

Our procedure consists of three steps: (i) we extract a candidate set of attributes $\mathcal{A}$ from captions in G_c using an LLM; (ii) we score each attribute $a \in \mathcal{A}$ on every instance $i \in G_c \cup G_r$ using the NLI-based entailment score $s(d_i, a)$; (iii) we select the DA a_t^* that maximizes the contrast between G_c and G_r:

$$a_t^* = \arg\max_{a \in \mathcal{A}} \left(\mathbb{E}_{i \in G_c}[s(d_i,a)] - \mathbb{E}_{j \in G_r}[s(d_j,a)] \right) \ .$$

The entailment score $s(d_i, a)$ is derived from the NLI logits. Detailed definitions of the scoring function and implementation settings are provided in Appendix C.

4.3.3 Attribute-guided group refinement

Although a_t^* is selected to maximize the contrast between G_c and G_r, the candidate attribute set $\mathcal{A}$ is extracted from G_c only, so a_t^* may also be possessed by candidates in G_r. We therefore refine the remainder set G_r by re-evaluating each candidate $i \in G_r$ via the NLI-based entailment score $s(d_i, a_t^*)$, and moving i to G_c if $s(d_i, a_t^*) \geq \tau$, where τ is a conservative threshold. This can add candidates to the core before asking the question. A partition is valid only when both groups are non-empty. If refinement produces an empty group, we try the next attribute in descending order of entailment gap. If no attribute yields two non-empty groups, we ask about the last attribute.

4.3.4 Interactive pruning and re-exploration

We then ask the user a binary question of whether the target T^* possesses the selected DA a_t^*. Based on the user's response, we update the active candidate pool as:

$$U_{t+1} = \begin{cases} G_c, & \text{if the user answers } \textbf{Yes}, \\ G_r, & \text{otherwise } (\textbf{No}), \end{cases} \tag{1}$$

thereby pruning the candidate pool according to the answer. Once the candidate pool U_{t+1} is finalized, we identify the target if $|U_{t+1}| = 1$. Otherwise, if $|U_{t+1}| \geq 2$, we proceed to the next RCJ round. When no attribute yields a valid split, a **Yes** answer leaves the pool unchanged.

When the user answers **No** but the remainder set is empty, *i.e.*, $|G_r| = 0$, the initial candidate pool may not contain the target. In this case, we resume the Pool Construction Stage to add one more candidate, then pre-prune the candidate pool with the target's facts collected from previous RCJ rounds before resuming RCJ.

5 Experiments

5.1 Benchmarks and implementation details

Benchmarks To demonstrate that ProCompNav is applicable to both ambiguous and detailed user queries, we evaluate it on two simulated benchmarks: CoIN-Bench [22] and text-goal navigation (TextNav) [21]. Both benchmarks feature multi-instance scenes but differ in initial user query specificity and the availability of user interaction. In CoIN-Bench, the agent is given only a coarse category at episode start, making interactive disambiguation necessary to identify the correct target instance; we follow its standard evaluation splits (*Val Seen*, *Val Seen Synonyms*, *Val Unseen*), and simulate user responses with an MLLM that has access to the image of the target instance. In contrast, in TextNav, the agent is given a detailed textual description of the target instance at episode start, yielding a non-interactive setting.

Evaluation metrics We report Success Rate (SR) and Success weighted by Path Length (SPL) as the primary evaluation metrics. SR measures the proportion of episodes in which the agent successfully reaches the target, while SPL evaluates exploration efficiency by comparing the agent's path length to the optimal path. In interactive settings, we additionally report Response Length (RL) and Number of Questions (NQ) as user-burden metrics: RL denotes the total response length per episode, measured as the token count of the user-simulator MLLM responses, and NQ averages questions over episodes with interaction to measure question burden when users are involved.

Implementation details We use a single Qwen3-VL-8B [1] model for both the text-only LLM and MLLM modules, and adopt VLFM [33] as the exploration backbone, consistent with AIUTA [22]. We initiate the Recursive Comparison Stage once the candidate pool size reaches $N_{\min} = 5$ on CoIN or $N_{\min} = 3$ on TextNav. If it has not started by step 400 on CoIN or step 600 on TextNav, we start it at that step. We use DeBERTa-v3-large [8, 12] as the NLI verifier for scoring attribute entailment in the Recursive Comparison Stage. We set the maximum episode length to 500 for CoIN following CoIN-Bench and to 1000 for TextNav following prior work. On CoIN, we use a four-question budget, except for Pooled Independent Matching (PIM), which has no question limit so it can gather information from every candidate as a strong baseline. An episode is considered successful if the agent executes `stop` within 1 m of the target instance. On CoIN, the Recursive Comparison Stage extracts attribute candidates from the collected candidate set U_t; on TextNav, we disable the user-question module and instead extract attributes directly from the detailed initial goal (see Appendix D). We further apply loop-escape and occasional 360° rotation heuristics (see Appendix E). We report the per-episode computational cost of AIUTA* and ProCompNav in Appendix O.

5.2 Main Results

5.2.1 Interactive Instance Navigation

As shown in Table 2, ProCompNav achieves the highest SR across all CoIN-Bench splits, outperforming both training-based methods [11, 21] and training-free methods that receive detailed target

Table 2: Performance on CoIN-Bench. In the *Judge* column, Indep/Comp indicate whether each method uses independent matching or comparative judgment for target disambiguation. The *Interact* column indicates whether the method interacts with the user for target disambiguation. For fair comparison across models, we include 3D-Mem* and AIUTA*, our reproductions using the same MLLM and LLM as ProCompNav. Results denoted by † are taken from [22] and those denoted by ‡ are taken from [10].

Method	Model Condition		Val Seen		Val Seen Synonyms		Val Unseen	
	Judge	Interact	SR ↑	SPL ↑	SR ↑	SPL ↑	SR ↑	SPL ↑
Training-based Methods								
Monolithic-GOAT† [11]	-	✗	6.6	3.1	13.1	6.5	0.2	0.1
PSL† [21]	-	✗	8.8	3.3	8.9	2.8	4.6	1.4
Training-free Methods (description input)								
3D-Mem* [29]	Comp	✗	15.9	11.0	18.9	13.6	5.7	3.9
Context-Nav‡ [10]	Indep	✗	13.5	6.7	20.3	10.9	11.3	5.2
Training-free Methods (category input)								
VLFM† [33]	Indep	✗	0.4	0.3	0.0	0.0	0.0	0.0
AIUTA† [22]	Indep	✓	7.4	2.9	14.4	8.0	6.7	2.3
AIUTA*	Indep	✓	10.5	4.5	15.3	8.4	8.9	4.0
Pooled Independent Matching	Indep	✓	17.5	5.1	22.0	8.1	13.3	5.1
ProCompNav (Ours)	Comp	✓	**23.7**	**7.0**	**28.1**	**8.5**	**17.0**	**6.2**

descriptions [10, 29], despite using only category-level input. On Val Seen, ProCompNav achieves a 126% relative SR improvement over AIUTA* [22]. Table 1 compares three end-to-end disambiguation strategies. Pooled Independent Matching uses the same pool construction stage as ProCompNav but applies independent matching, improving SR over AIUTA*. This suggests that collecting candidates before making a decision helps avoid premature selection of early-encountered distractors. However, because its questions are generated independently for each candidate, the collected information may not distinguish the target from distractors that share similar attributes. Using the same pool construction stage, ProCompNav further improves SR by selecting attribute-value pairs that split the pool and pruning candidates from binary feedback. ProCompNav also achieves lower RL and NQ than Pooled Independent Matching, consistent with its binary questions and selection of attributes that distinguish the current candidates.

5.2.2 Non-interactive Instance Navigation

ProCompNav achieves the highest SR (28.5%) among the evaluated TextNav methods, an 8.8% relative improvement over Context-Nav [10] (Table 3). This result shows that candidate-relative comparison remains effective with detailed target descriptions and without predefined goal-to-scene matching rules [10, 32]. We evaluate this non-interactive setting on TextNav [21]. Instead of asking user questions, ProCompNav extracts attributes from the initial target description. It otherwise retains the collect-and-compare procedure: it constructs a candidate pool, selects goal-derived attributes from candidate contrasts, refines candidate groups, and recursively retains the group supported by each attribute. Thus, user feedback is removed while the core candidate-comparison mechanism remains unchanged (Appendix D).

Table 3: Performance on TextNav. In the *Judgment* column, Indep/Comp indicate whether each method uses independent matching or comparative judgment for target disambiguation. Results denoted by † are taken from [32], those denoted by ‡ are taken from [10]. Methods denoted by * are our reproductions using the same MLLM and LLM as ProCompNav.

Method	Judgment	SR ↑	SPL ↑
Training-based Methods			
Modular-GOAT† [11]	-	17.0	8.8
PSL† [21]	-	16.5	7.5
Training-free Methods			
UniGoal† [32]	Indep	20.2	**11.4**
AIUTA* [22]	Indep	10.9	2.7
3D-Mem* [29]	Comp	14.1	9.6
Context-Nav‡ [10]	Indep	26.2	9.1
ProCompNav (Ours)	Comp	**28.5**	6.9

5.2.3 Robustness to User Feedback

Table 4: Results with changed user feedback on CoIN-Bench Val Seen. p is the probability of IDK or flipped feedback. Original reports the reference values from Table 1.

Method	Noise	Original			$p = 10\%$			$p = 20\%$		
		SR ↑	RL ↓	NQ ↓	SR ↑	RL ↓	NQ ↓	SR ↑	RL ↓	NQ ↓
AIUTA*	Flip	10.5	109.5	1.2	11.3	99.3	1.2	11.3	107.6	1.2
	IDK				10.2	101.8	1.4	12.4	89.7	1.6
	Combined				9.9	98.5	1.4	10.2	90.3	1.6
Pooled	Flip	17.5	460.2	4.1	16.6	460.9	4.2	16.9	463.5	4.2
	IDK				16.4	411.6	4.5	17.1	360.5	4.9
	Combined				17.7	421.4	4.5	16.0	374.2	5.1
ProCompNav	Flip	**23.7**	**4.2**	2.2	**19.9**	**4.2**	2.2	**17.1**	**4.1**	2.2
	IDK				**20.8**	**4.7**	2.5	**21.2**	**4.1**	2.8
	Combined				**20.2**	**4.1**	2.4	**17.3**	**4.0**	2.7

The MLLM user simulator is not an idealized noise-free user. We model uncertain or incorrect feedback with IDK, Flip, and Combined perturbations at $p \in \{10\%, 20\%\}$. IDK returns "I don't know," while Flip reverses ProCompNav's binary feedback and changes only the queried attribute for the open-ended baselines (Appendix T). Combined applies IDK with probability p and otherwise applies Flip with probability p. ProCompNav retains the highest SR in every tested condition, including $p = 20\%$ (Table 4). IDK mainly increases NQ as RCJ proceeds to the next-ranked attribute. Flip causes a larger SR drop because each binary answer determines which group is pruned, whereas the baselines retain other information in their free-form responses.

5.3 Effect of Design Choices

Table 5 reports the effect of five design choices on the CoIN-Bench Val Seen split. Disabling multi-view aggregation and using only a single view per candidate drops SR from 23.7 to 23.1. Removing NLI entirely and letting the LLM alone pick a discriminative attribute from the two group descriptions drops SR to 20.6, showing that adding the entailment LM provides further performance gain on top of the LLM. Replacing our core-remainder split with KMeans grouping based on the same averaged text-image similarity reduces SR from 23.7% to 20.6%, supporting the effectiveness of our similarity-based core selection. Disabling the refinement module drops SR to 21.8: since the discriminative attribute is selected from the core set G_c only, the remainder group may still contain candidates that match it, and refinement is needed to recover them. The prompt-only variant selects an attribute and splits the pool in one LLM call but achieves lower SR. Further question-format and candidate-comparison ablations are in Appendix G.

Table 5: Ablations on CoIN-Bench Val Seen.

Configuration	SR ↑	SPL ↑	RL ↓	NQ ↓
w/o multi-view aggregation	23.1	6.3	4.2	2.1
LLM-only DA (w/o NLI)	20.6	5.5	4.2	2.2
w/o similarity-based core set	20.6	5.8	4.5	2.4
w/o refinement	21.8	6.0	4.2	2.1
Prompt-only	20.9	5.9	4.7	2.5
ProCompNav (Ours)	**23.7**	**7.0**	4.2	2.2

5.4 Human Evaluation of Interaction Strategies

Human users favored the binary-question condition. In Study 1, all 20 participants preferred and responded faster in the binary condition, while 89.1% of Korean open-ended responses contained at most three words. Compared with the lengthy MLLM-simulated open-ended responses in Table 1, this reveals a simulator-human verbosity mismatch in open-ended interaction. Binary questions largely avoid this mismatch because simulated and real users share the same constrained response format. This makes binary interaction better aligned between simulated evaluation and real-user deployment. In Study 2, 14 of 20 participants (70%) preferred batched delivery, consistent with prior work on interruption costs and notification batching [2, 6]. See Appendix R for human evaluation details.

Table 6: Human evaluation of response format (Study 1) and question delivery (Study 2). Response time and Preference denote the median per question and participant counts, respectively.

	Study 1		*Study 2*	
Metric	Open-ended	Binary	Interleaved	Batched
Response time ↓	9.25 s	**2.69 s**	6.27 s	**5.28 s**
Preference ↑	0/20	**20/20**	6/20	**14/20**

6 Analysis

6.1 Termination-Step Analysis

To demonstrate the advantage of our collect-then-compare strategy, we compare the episode termination steps and success rates of AIUTA, Pooled Independent Matching, and ProCompNav on the CoIN-Bench Val Seen split in Figure 3. AIUTA, which adopts an independent matching strategy, either makes a premature wrong target decision upon encountering a distractor (see the 1-99 step bin in Figure 3), or fails to make any target decision until the episode horizon is exhausted (see the 500 (max) step bin in Figure 3), both of which lead to a low success rate. Pooling reduces early commitments, while comparative judgment improves decisions after candidate collection. Relative to AIUTA, Pooled Independent Matching shifts terminations away from the first 100 steps. ProCompNav has a similar termination distribution to Pooled Independent Matching but accumulates more successes.

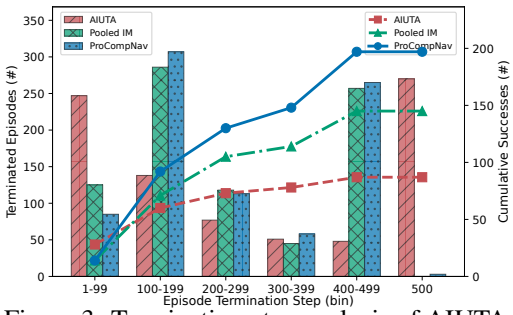


Figure 3: Termination-step analysis of AIUTA, Pooled Independent Matching, and ProCompNav. Bars show terminated episodes in 100-step bins on the left axis, and lines show cumulative successful episodes on the right axis. The final bin denotes the 500-step horizon.

6.2 Failure Analysis

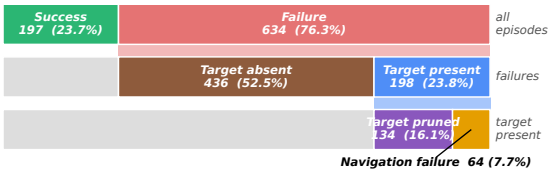


Figure 4: Failure breakdown on CoIN-Bench Val Seen. Percentages are over all 831 episodes.

Table 7: Possible causes of target pruning on CoIN-Bench Val Seen ($n = 134$). Shares are within this subset.

Category	n	Share (%)
Caption Hallucination	24	17.9
Caption Omission	48	35.8
Core-set False Positive	24	17.9
Refinement False Negative	3	2.2
Refinement False Positive	0	0.0
Simulator-Judge Mismatch	35	26.1
Total	134	100.0

Figure 4 shows that target absence is the dominant failure mode, accounting for 436 of 634 failures (68.8%). These cases occur during VLFM-based exploration and object detection, before RCJ. Among the 198 target-present failures, the agent selects the target but fails during navigation in 64 episodes. In the remaining 134, the target is pruned during RCJ.

To investigate why targets are pruned during RCJ, we examine these 134 episodes using Gemini 3 Flash Preview[1] for post-hoc judgments (Table 7). The dominant factors (53.7%) are hallucination and attribute omission during caption generation from each candidate's multi-view images in the Pool Construction Stage. In 17.9% of cases, the target is in the core set, but Gemini judges that it does not have the selected attribute. A **No** answer then removes the core set, including the target. Only 2.2% are categorized as refinement false negatives, with no false positives observed at the conservative threshold $\tau = 0.9$. A further 26.1% involve Qwen3-VL-8B giving a different answer from Gemini's judgment. Even for a simple yes/no question, answers can depend on subjective interpretation (e.g., whether a shade is considered light brown). Appendix S details the analysis procedure.

6.3 Qualitative Analysis

Figure 5 contrasts independent matching and ProCompNav on the ambiguous query "Find the dresser". The independent matching baseline [22] (left) asks an open-ended attribute question ("What is the color and material of the handles…"), forcing a long descriptive response. Even with this detailed answer, the baseline often decides on the first detected distractor as the target. ProCompNav

[1] https://blog.google/products-and-platforms/products/gemini/gemini-3-flash/

(right) instead defers its decision, builds a candidate pool, and asks discriminative binary questions. In Round 1, "Is there a red box nearby?" splits the pool into a core set (G_c) and a remainder set (G_r), and a single Yes eliminates three distractors. In Round 2, "Is there a TV on top?" isolates the true target (T^*).

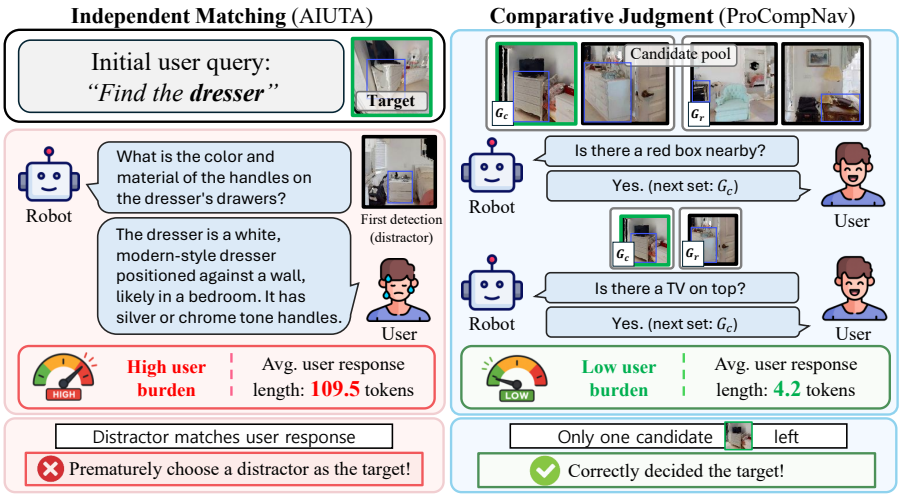


Figure 5: Qualitative comparison of Independent Matching and Comparative Judgment under a minimally specified query ("Find the dresser"). **Independent Matching** (left) asks the user to describe an attribute of the intended target without knowing whether that attribute distinguishes it from distractors. This burdens the user with a long response and can lead the robot to mistakenly match a distractor sharing the same attribute. In contrast, **Comparative Judgment** (right) asks about discriminative attributes in a way that elicits only short user responses, reducing verbal burden while correctly isolating the user-intended target.

7 Conclusion

We studied proactive clarification in interactive instance navigation under ambiguous user requests. We proposed ProCompNav, which first constructs a candidate pool and then identifies the target through Recursive Comparative Judgment. Comparing candidates lets the robot select both an attribute and a value to split the pool, without requiring an attribute unique to the target. This makes binary questions useful for eliminating candidates, beyond merely shortening responses. On CoIN-Bench, ProCompNav improves Success Rate while reducing response length, and it achieves the highest Success Rate among evaluated methods on non-interactive TextNav. In two human studies, participants preferred binary questions to open-ended questions and batched delivery to interleaved interaction. Together, these results suggest that agentic robots should be designed for both task performance and user-friendly interaction.

Limitations ProCompNav is currently evaluated in simulation, and extending it to real-world robotic platforms remains an important future direction. ProCompNav relies on its exploration and detection modules to place the target in the candidate pool, leaving room for improvement with stronger components. It currently does not recover the target once it is mistakenly pruned by incorrect binary feedback, motivating future work on recovery mechanisms. Finally, requiring a minimum pool size may be inefficient when few same-category instances exist, motivating dynamic adaptation.

Acknowledgments and Disclosure of Funding

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (RS-2019-II191906, Artificial Intelligence Graduate School Program (POSTECH); IITP-2026-RS-2024-00437866, Information Technology Research Center (ITRC); IITP-2026-RS-2026-25616370, AI Star Fellowship Support Program), and by a Korea Institute for Advancement of Technology (KIAT) grant funded by the Ministry of Trade, Industry and Energy (MOTIE), Republic of Korea (RS-2025-00564342). The authors declare no competing interests.

References

[1] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[2] Brian P. Bailey and Joseph A. Konstan. On the need for attention-aware systems: Measuring effects of interruption on task performance, error rate, and affective state. *Computers in Human Behavior*, 22(4):685–708, 2006. doi: 10.1016/j.chb.2005.12.009.

[3] Wenzhe Cai, Siyuan Huang, Guangran Cheng, Yuxing Long, Peng Gao, Changyin Sun, and Hao Dong. Bridging zero-shot object navigation and foundation models through pixel-guided navigation skill. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5228–5234. IEEE, 2024.

[4] Yihan Cao, Jiazhao Zhang, Zhinan Yu, Shuzhen Liu, Zheng Qin, Qin Zou, Bo Du, and Kai Xu. Cognav: Cognitive process modeling for object goal navigation with llms. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9550–9560. IEEE, 2025.

[5] Eugenio Chisari, Jan Ole Von Hartz, Fabien Despinoy, and Abhinav Valada. Robotic task ambiguity resolution via natural language interaction. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 14821–14827. IEEE, 2025.

[6] Nicholas Fitz, Kostadin Kushlev, Ranjan Jagannathan, Terrel Lewis, Devang Paliwal, and Dan Ariely. Batching smartphone notifications can improve well-being. *Computers in Human Behavior*, 101:84–94, 2019.

[7] Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23171–23181, 2023.

[8] Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

[9] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. In *Conference on Robot Learning*, pages 1769–1782. PMLR, 2023.

[10] Won Shik Jang and Ue-Hwan Kim. Context-nav: Context-driven exploration and viewpoint-aware 3d spatial reasoning for instance navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9626–9636, 2026.

[11] Mukul Khanna, Ram Ramrakhya, Gunjan Chhablani, Sriram Yenamandra, Theophile Gervet, Matthew Chang, Zsolt Kira, Devendra Singh Chaplot, Dhruv Batra, and Roozbeh Mottaghi. Goat-bench: A benchmark for multi-modal lifelong navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16373–16383, 2024.

[12] Moritz Laurer, Wouter Van Atteveldt, Andreu Casas, and Kasper Welbers. Less annotating, more classifying: Addressing the data scarcity issue of supervised machine learning with deep transfer learning and bert-nli. *Political Analysis*, 32(1):84–100, 2024.

[13] Xingyao Lin, Xinghao Zhu, Tianyi Lu, Sicheng Xie, Hui Zhang, Xipeng Qiu, Zuxuan Wu, and Yu-Gang Jiang. Ask-to-clarify: Resolving instruction ambiguity through multi-turn dialogue. *arXiv preprint arXiv:2509.15061*, 2025.

[14] Yuxing Long, Wenzhe Cai, Hongcheng Wang, Guanqi Zhan, and Hao Dong. Instructnav: Zero-shot system for generic instruction navigation in unexplored environment. *arXiv preprint arXiv:2406.04882*, 2024.

[15] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[16] Pradip Pramanick, Chayan Sarkar, Sayan Paul, Ruddra dev Roychoudhury, and Brojeshwar Bhowmick. Doro: Disambiguation of referred object for embodied agents. *IEEE Robotics and Automation Letters*, 7(4):10826–10833, 2022.

[17] Qwen Team. Qwen3.6-Plus: Towards real world agents, April 2026. URL https://qwen.ai/blog?id=qwen3.6.

[18] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-Matterport 3D Dataset (HM3D): 1000 large-scale 3d environments for embodied AI. In *Proceedings of the NeurIPS Datasets and Benchmarks Track*, 2021. URL https://arxiv.org/abs/2109.08238.

[19] Allen Z Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, et al. Robots that ask for help: Uncertainty alignment for large language model planners. In *Conference on Robot Learning*, pages 661–682. PMLR, 2023.

[20] Yijia Shao, Vinay Samuel, Yucheng Jiang, John Yang, and Diyi Yang. Collaborative gym: A framework for enabling and evaluating human-agent collaboration. In *International Conference on Learning Representations*, volume 2026, pages 99616–99649, 2026.

[21] Xinyu Sun, Lizhao Liu, Hongyan Zhi, Ronghe Qiu, and Junwei Liang. Prioritized semantic learning for zero-shot instance navigation. In *European Conference on Computer Vision*, pages 161–178. Springer, 2024.

[22] Francesco Taioli, Edoardo Zorzi, Gianni Franchi, Alberto Castellini, Alessandro Farinelli, Marco Cristani, and Yiming Wang. Collaborative instance object navigation: Leveraging uncertainty-awareness to minimize human-agent dialogues. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18781–18792, 2025.

[23] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

[24] Wenxuan Wang, Shi Juluan, Zixuan Ling, Yuk-Kit Chan, Chaozheng Wang, Cheryl Lee, Youliang Yuan, Jen-tse Huang, Wenxiang Jiao, and Michael R Lyu. Learning to ask: When llm agents meet unclear instruction. In *Proceedings of the 2025 conference on empirical methods in natural language processing*, pages 21784–21795, 2025.

[25] Yiyao Wang, Sixian Zhang, Keming Zhang, Xinhang Song, Songjie Du, and Shuqiang Jiang. Trajrag: Retrieving geometric-semantic experience for zero-shot object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15166–15176, 2026.

[26] Pengying Wu, Yao Mu, Bingxian Wu, Yi Hou, Ji Ma, Shanghang Zhang, and Chang Liu. Voronav: Voronoi-based zero-shot object navigation with large language model. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 53757–53775. PMLR, 2024.

[27] Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. Collabllm: From passive responders to active collaborators. *arXiv preprint arXiv:2502.00640*, 2025.

[28] Yang Yang, Xibai Lou, and Changhyun Choi. Interactive robotic grasping with attribute-guided disambiguation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8914–8920. IEEE, 2022.

[29] Yuncong Yang, Han Yang, Jiachen Zhou, Peihao Chen, Hongxin Zhang, Yilun Du, and Chuang Gan. 3d-mem: 3d scene memory for embodied exploration and reasoning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 17294–17303, 2025.

[30] Zebin Yang, Sunjian Zheng, Tong Xie, Tianshi Xu, Bo Yu, Fan Wang, Jie Tang, Shaoshan Liu, and Meng Li. Efficientnav: Towards on-device object-goal navigation with navigation map caching and retrieval. *Advances in Neural Information Processing Systems*, 38:4286–4312, 2025.

[31] 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.

[32] Hang Yin, Xiuwei Xu, Linqing Zhao, Ziwei Wang, Jie Zhou, and Jiwen Lu. Unigoal: Towards universal zero-shot goal-oriented navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19057–19066, 2025.

[33] Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 42–48. IEEE, 2024.

[34] Bangguo Yu, Hamidreza Kasaei, and Ming Cao. L3mvn: Leveraging large language models for visual target navigation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3554–3560. IEEE, 2023.

[35] Shuaihang Yuan, Hao Huang, Yu Hao, Congcong Wen, Anthony Tzes, and Yi Fang. Gamap: Zero-shot object goal navigation with multi-scale geometric-affordance guidance. *Advances in Neural Information Processing Systems*, 37:39386–39408, 2024.

[36] Lingfeng Zhang, Qiang Zhang, Hao Wang, Erjia Xiao, Zixuan Jiang, Honglei Chen, and Renjing Xu. Trihelper: Zero-shot object navigation with dynamic assistance. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10035–10042. IEEE, 2024.

[37] Lingfeng Zhang, Hao Wang, Erjia Xiao, Xinyao Zhang, Qiang Zhang, Zixuan Jiang, and Renjing Xu. Multi-floor zero-shot object navigation policy. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6416–6422. IEEE, 2025.

[38] Sixian Zhang, Xinyao Yu, Xinhang Song, Yiyao Wang, and Shuqiang Jiang. Function-centric bayesian network for zero-shot object goal navigation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19535–19545. IEEE, 2025.

[39] Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. Esc: Exploration with soft commonsense constraints for zero-shot object navigation. In *International Conference on Machine Learning*, pages 42829–42842. PMLR, 2023.

[40] Zibo Zhou, Yue Hu, Lingkai Zhang, Zonglin Li, and Siheng Chen. Beliefmapnav: 3d voxel-based belief map for zero-shot object navigation. *Advances in Neural Information Processing Systems*, 38:83008–83036, 2025.

[41] Filippo Ziliotto, Tommaso Campari, Luciano Serafini, and Lamberto Ballan. Tango: training-free embodied ai agents for open-world tasks. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 24603–24613, 2025.

A Baselines

For CoIN-Bench [22], we compare ProCompNav against trained policies **Monolithic-GOAT** [11] and **PSL** [21]. Among training-free methods, **3D-Mem** [29] and **Context-Nav** [10] receive a detailed target description—a richer input than the target category available to ProCompNav—while **VLFM** [33] and **AIUTA** [22] receive only the target category. VLFM navigates without interaction, whereas AIUTA asks the user simulator open-ended clarifying questions. We further evaluate on TextNav to verify that ProCompNav generalizes to the non-interactive, detailed-description setting, using the same baselines as CoIN-Bench and additionally including **UniGoal** [32]. We include **AIUTA***, our reproduction of AIUTA with the same LLM and MLLM as ProCompNav; on TextNav, it is additionally adapted for the non-interactive setting (see Appendix F). **Pooled independent matching** is a baseline we implement, pairing ProCompNav's pool construction stage with AIUTA's independent matching. Once the pool is collected, the agent visits the candidates in two passes. In the first pass, it asks the user one open-ended question per candidate, conditioning each question on the facts accumulated from the previous candidates so that each new question elicits information not yet gathered. In the second pass, an LLM scores how consistent each candidate's description is with the final fact set. When multiple candidates are tied at the top score, an LLM is given their descriptions and asked to select the one most consistent with the accumulated facts.

B Pool Construction Stage Details

Cross-view instance assignment. For each new detection of category-c, the agent back-projects the bounding box to a 3D point cloud and computes the symmetric overlap ratio against every existing candidate's accumulated point cloud $\mathcal{P}_i$, defined as the fraction of points in either cloud that have a neighbor in the other within radius $\epsilon = 0.03\,\text{m}$ (we take the maximum of the two directed fractions). If the maximum overlap is at least 0.3, the detection is assigned to the matching candidate and its RGB view is added to $\mathcal{V}_i$. Otherwise, a new candidate is initialized.

Diverse view selection. For each candidate i, we extract a visual embedding for every view $v \in \mathcal{V}_i$ using DINOv2 [15], cluster the embeddings into $K = 6$ clusters via K-means, and select the view closest to each centroid as a representative. We arrange the resulting representatives into the multi-view collage I_i. An MLLM filters out views in which the candidate is not clearly visible to reduce noise in caption generation. To base RCJ's visual similarity on a clear observation, the MLLM selects the clearest retained view for the image embedding. Figure 8 connects recorded candidate views to generated descriptions used as RCJ input.

C NLI-based Entailment Scoring

We instantiate the attribute verifier with a pretrained Natural Language Inference (NLI) classifier. For each instance description d_i (premise) and candidate attribute a, we form the hypothesis "the instance has attribute a" and query the NLI model to obtain logits $\ell_E(d_i, a)$, $\ell_N(d_i, a)$, and $\ell_C(d_i, a)$ for entailment, neutral, and contradiction, respectively. We convert these logits into a scalar entailment score

$$s(d_i, a) = \sigma\big(\ell_E(d_i, a) - \max(\ell_N(d_i, a), \ell_C(d_i, a))\big)\ ,$$

where σ is the sigmoid function. This score provides a calibrated measure of attribute support that is comparable across instances and attributes. In our implementation, we use a lightweight NLI backbone (*e.g.*, DeBERTa-v3-large [8, 12]) and apply the same scoring to all candidates in G_c and G_r.

D Adaptation of ProCompNav to TextNav

Unlike CoIN-Bench, TextNav provides a detailed target description at the beginning of each episode and does not allow user interaction. We use $N_{\min} = 3$ on TextNav and start the Recursive Comparison Stage at step 600 if it has not started earlier. We disable the question-asking module and adapt the Recursive Comparison Stage in the following two ways.

(1) Goal-derived attribute set. Instead of extracting attribute candidates from the current core set G_c at each iteration, we parse a fixed attribute set $\mathcal{A}$ once from the text goal describing the target

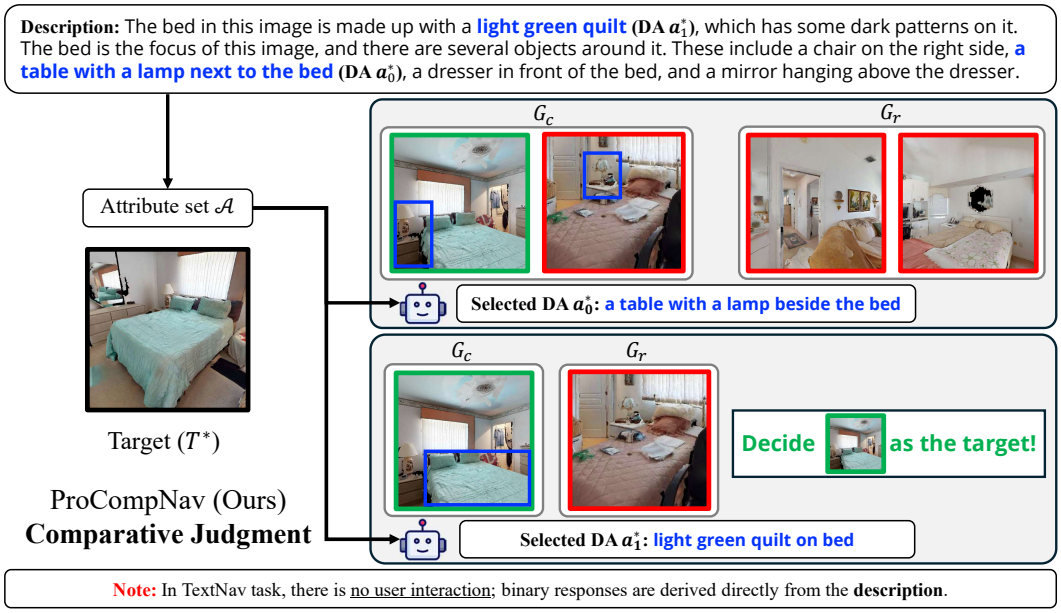


Figure 6: TextNav adaptation of the Recursive Comparison Stage. In TextNav, ProCompNav pre-extracts an attribute set $\mathcal{A}$ from the text goal before the Recursive Comparison Stage and, at each round, selects the discriminative attribute that best supports the current core set G_c. Since these attributes are derived from the goal, the active set is always updated to G_c. The agent terminates once the target-selection condition is met.

instance using the LLM. Since $\mathcal{A}$ is derived from the goal description, its attributes are assumed to hold for the target. Accordingly, after attribute-guided group refinement, we always set $U_{t+1} = G_c$, rather than choosing between G_c and G_r based on user feedback as in CoIN.

(2) Candidate verification without user feedback. When $G_r = \emptyset$, we invoke a text-only LLM-based verifier. Specifically, if $|U_t| = 1$, we verify the sole remaining candidate instance; if $G_r = \emptyset$ and $|G_c| \geq 2$, we verify the candidate instance $i \in G_c$ with the highest entailment score $s(d_i, a_t^*)$. The verifier compares the candidate description d_i with the text goal describing the target instance and returns a binary accept/reject decision. If accepted, the agent executes `stop`; otherwise, it resumes the Pool Construction Stage to collect additional candidates from unexplored regions.

E Adapting Object Navigation for Instance-Level Exploration

Language-driven instance navigation methods often adopt an object navigation (ON) backbone for exploration—*e.g.*, AIUTA [22] uses VLFM [33] and UniGoal [32] uses SG-Nav [31]. These methods repeatedly select the frontier with the highest reasoning score and are designed to reach *any* instance of the target category. In instance navigation, however, the first observed instance is not necessarily the target, so the agent may need to continue exploring even after an initial detection.

Directly extending ON backbones to this setting introduces two practical issues. First, greedy top-1 frontier selection can repeatedly favor an unreachable or suboptimal frontier, causing local loops and reducing exploration coverage. Second, with a forward-facing monocular camera and a narrow field of view (30°), the agent may pass nearby objects without observing them, leaving potential candidates undetected. To address these issues, we add two lightweight heuristics to our VLFM-based exploration backbone.

E.1 Loop detection and frontier escape

We maintain an exponential moving average (EMA) of the agent's position to detect trajectory stagnation. Let $\mathbf{p}_t$ denote the agent's position at step t. The trajectory center $\mathbf{c}_t$ and spread s_t are

updated as:

$$\mathbf{c}_t = \alpha\, \mathbf{p}_t + (1-\alpha)\, \mathbf{c}_{t-1}\ ,$$
$$s_t = \alpha\, \|\mathbf{p}_t - \mathbf{c}_t\|^2 + (1-\alpha)\, s_{t-1}\ .$$

We then define the loopness score as

$$\mathrm{loop}_t = \exp\left(-\frac{\|\mathbf{p}_t - \mathbf{c}_t\|^2}{s_t}\right)\ ,$$

which approaches 1 when the agent remains near its recent trajectory center. The normalization by s_t makes the score adaptive to the recent motion scale. If $\mathrm{loop}_t \geq 0.9$ persists for 5 consecutive steps while moving toward a frontier, we temporarily blacklist all frontiers in the same grid cell by assigning them a large negative score, forcing the planner to redirect exploration elsewhere.

E.2 Line-of-sight rotation

To compensate for the narrow field of view, we trigger an in-place 360° rotation when the surroundings are sufficiently open. We define the *openness* of the current position (x_0, y_0) as the fraction of 360 uniformly spaced angular bins (1° resolution) whose line of sight is not blocked on the occupancy map:

$$\mathrm{openness} = 1 - \frac{|\{\text{occluded bins}\}|}{360}\ .$$

A bin at angle θ is marked occluded if the ray in that direction intersects an obstacle, where each obstacle point (x_i, y_i) is associated with angle $\theta_i = \mathrm{atan2}(y_i - y_0,\, x_i - x_0)$. We execute a 360° rotation when $\mathrm{openness} \geq 0.1$ and the agent is at least 1.0 m away from the previous rotation point, which avoids redundant rotations in confined areas.

F Adaptation of AIUTA to TextNav

For a fair comparison under the same model condition, we adapt AIUTA [22] to both CoIN-Bench [22] and TextNav [21] using the same Qwen3-VL-8B [1] MLLM as in our method. On CoIN-Bench, AIUTA* follows the original interactive setting. On TextNav, however, we remove the user-interaction module and replace the original dynamically updated fact memory with the given detailed text goal, which is kept fixed throughout the episode. Accordingly, when a target-category object is detected, AIUTA* no longer considers the `Ask` action; instead, it compares the detected object against the fixed text goal and chooses only between `Stop` and `Skip`. If the detected object is judged to match the detailed goal, the agent executes `stop`; otherwise, it skips the instance and continues exploration.

G Question Format and Candidate Comparison

Table 8: Question format and candidate comparison on all 831 CoIN-Bench Val Seen episodes, using the same pool construction stage and Qwen3-VL-8B backbone. Pooled Independent Matching (PIM) and Binary PIM accumulate target facts and score candidates independently. The ablation variants "No candidate information" and "Single candidate" prune candidates using binary feedback, as in ProCompNav's RCJ. MaxQ is the percentage of all episodes reaching the four-question limit. NQ averages over interactive episodes.

Method	Question	Attribute selection	SR↑	SPL↑	RL↓	NQ↓	MaxQ↓
Question Format (per-candidate questioning)							
Pooled Independent Matching (PIM)	Open-ended	Per candidate	17.5	5.1	460.2	4.1	-
Binary PIM	Binary	Per candidate	17.2	5.2	7.5	4.3	-
Attribute Selection (binary questions)							
No candidate information	Binary	No candidate captions	14.1	3.6	5.7	3.0	56
Single candidate	Binary	One candidate caption	19.6	5.7	5.2	2.7	31
ProCompNav	Binary	Structured comparison	**23.7**	**7.0**	**4.2**	**2.2**	**2**

Question format. We compare Pooled Independent Matching (PIM) and Binary PIM to verify the response-length benefit of binary questions. Both ask one question per candidate without a

four-question limit and retain fact accumulation and independent candidate scoring. Binary PIM replaces PIM's open-ended questions with binary attribute-value questions. Its LLM receives only the current candidate's description and accumulated target facts, without other candidate descriptions. Simulated RL decreases from 460.2 to 7.5 tokens while SR remains comparable, confirming the response-length reduction from binary questions under the same per-candidate questioning strategy.

Attribute selection. We next vary how attribute-value pairs are selected for binary questions under the main setting's four-question budget (Section 5.1). "No candidate information" follows a twenty-questions-style interaction: the first question uses only the target category (e.g., sofa), and subsequent questions use previous questions and user responses. "Single candidate" additionally receives one candidate caption per question, selected sequentially as in PIM and Binary PIM. This is intended to avoid implicit candidate comparison that could arise from presenting all captions together. It takes the next unused caption among the remaining candidates in collection order. Both include previously queried attributes in the prompt for each new question and instruct the LLM not to reuse them.

After each answer, candidates inconsistent with it are pruned. The two variants use the same NLI verifier and entailment-score threshold (0.9) as ProCompNav, including filtering after re-exploration and entailment-score-based selection at the question limit. Questioning ends when one candidate remains or the four-question budget is exhausted. In the latter case, a candidate is selected using entailment scores. If no candidates remain after pruning, we apply the re-exploration and fallback rules. Thus, unlike per-candidate questioning, NQ depends on how effectively each question reduces the pool. These ablations examine how attribute-value choices that do not effectively narrow the candidate pool affect RL and NQ, even with binary responses.

Beyond Binary Format. Binary questions can greatly reduce response length relative to open-ended questions, but this format alone does not ensure fewer questions. "Single candidate", which receives one candidate caption per question, improves SR and reduces NQ to 2.7 and MaxQ to 31% compared with "No candidate information". However, without identifying a value that clearly eliminates candidates, binary questioning can still require many questions to cover possible values. ProCompNav compares candidates to choose both the attribute and the value to verify, achieving higher SR with NQ of 2.2 and MaxQ of 2% (Table 8).

H Increasing Query Specificity

Table 9: Performance with increasingly specific CoIN queries. k is the number of target hints. All results use the 621 Val Seen episodes whose target descriptions yield at least three parsed atomic hints. †The $k = 0$ reference is also aggregated over these 621 episodes, rather than the full Val Seen split.

	$k = 0^{\dagger}$			$k = 1$			$k = 2$			$k = 3$		
Method	SR ↑	RL ↓	NQ ↓	SR ↑	RL ↓	NQ ↓	SR ↑	RL ↓	NQ ↓	SR ↑	RL ↓	NQ ↓
AIUTA*	12.2	116.1	1.3	10.1	16.2	2.0	6.8	5.5	1.1	8.5	8.9	1.2
Pooled	16.6	430.7	3.9	19.3	421.1	3.8	17.2	455.9	4.0	19.2	452.6	4.1
ProCompNav	**21.9**	**4.1**	2.2	**21.9**	**4.2**	2.3	**21.7**	**4.3**	2.3	**22.4**	**4.2**	2.3

We use Qwen3.6-Plus [17] for atomic hint extraction. We evaluate all $k \in \{0, 1, 2, 3\}$ on the same 621 Val Seen episodes whose target descriptions yield at least three parsed atomic hints. We augment the category-only CoIN query with k hints, which ProCompNav applies before extracting new DAs and querying the user. ProCompNav achieves the highest SR at every k, but SR and NQ do not improve monotonically as hints are added. An atomic target fact may also hold for distractors, so more hints do not necessarily resolve ambiguity.

Figure 7 illustrates the nested hint inputs alongside their source target descriptions. The category query stays fixed, and the separate hint list grows by one atomic statement at each k.

(a) Cabinet Episode 1160

Target | Distractor 1 | Distractor 2



Query: "Find the cabinet"

Source description (GOAT-Bench)

white cabinet with an owl painting on it located to the right of the box and below the painting.

k	Hints available at episode start
0	None
1	**is below the painting**
2	is below the painting **is located to the right of the box**
3	is below the painting is located to the right of the box **is a white cabinet**

(b) Bed Episode 553

Target | Distractor 1 | Distractor 2



Query: "Find the bed"

Source description (GOAT-Bench)

bed with purple and yellow pillows. the bed is located next to a board and there is a picture nearby. look for the bed within the region where the pillow is present.

k	Hints available at episode start
0	None
1	**is located near a picture**
2	is located near a picture **has yellow pillows**
3	is located near a picture has yellow pillows **is located next to a board**

Figure 7: Atomic hints extracted from GOAT-Bench [11] target descriptions provided in CoIN-Bench. Gray boxes reproduce the source descriptions. Each hint specifies one target property or relation. The category query is unchanged for all k; hints are supplied separately. Each row retains the previous hints and adds the bold blue hint. Hint wording is preserved. Green and red borders indicate the target and two selected same-category distractors per episode. Images are re-rendered for illustration, not agent observations.

I Core-Set Selection

Table 10: Core-set selection on CoIN-Bench Val Seen. NQ averages questions over episodes with interaction, and MaxQ denotes the percentage of all episodes reaching the four-question limit. Stage 2 time is the average total time spent in the Recursive Comparison Stage per episode, excluding pool construction and navigation.

Method	SR↑	SPL↑	RL↓	NQ↓	MaxQ↓	Stage 2 time↓
w/o core set (KMeans)	20.6	5.8	4.5	2.4	9%	10.8 s
ProCompNav	**23.7**	**7.0**	**4.2**	**2.2**	**2%**	**6.3 s**

Setup. We replace core selection with KMeans ($k = 2$, `random_state=42`, `n_init=10`), using the same 831 Val Seen episodes and RTX 3090 GPUs. KMeans concatenates separately ℓ_2-normalized text and image embeddings, each scaled by $1/\sqrt{2}$, matching ProCompNav's equally weighted similarity. Pool construction, NLI attribute selection, refinement, the user simulator, and the four-question budget remain unchanged.

Results. Compared with KMeans using the same text and image features, ProCompNav achieves higher SR and lower RL, NQ, and MaxQ. It also takes less time in Stage 2 (Table 10).



Figure 8: Multi-view observations and generated captions. Each panel shows recorded views of one candidate and part of its multi-view generated caption. Highlighted words describe details visible in the images. The generated captions are used as input to RCJ.

Table 11: Ablation on refinement threshold on Val Seen.

Threshold	SR ↑	SPL ↑	RL ↓	NQ ↓
No refinement	21.8	6.0	4.2	2.1
$\tau = 0.5$	22.0	6.3	4.1	2.2
$\tau = 0.7$	22.6	6.5	4.2	2.2
$\tau = 0.9$ (ours)	23.7	7.0	4.2	2.2

J Effect of The Refinement Threshold

Table 11 shows that refinement improves SR and SPL over the no-refinement variant with almost no increase in user burden. Among the refinement thresholds, the conservative setting $\tau = 0.9$ performs best, achieving the highest SR and SPL while keeping RL and NQ nearly unchanged. This suggests that refinement is beneficial, but should be applied conservatively to avoid introducing unreliable comparisons.

Core-set sizes. Across 1,752 grouping rounds, all 1,130 rounds with at least three candidates end in a core pair, while the remaining 622 two-candidate rounds use a singleton core. Refinement expands the core in 139 rounds, increasing its mean size from 2.00 to 2.13. The final partition contains two non-empty groups in 1,128 of the 1,130 peeling rounds.

Refinement example. Figure 9 shows how refinement preserves a target initially placed in the remainder set despite sharing the selected attribute with the core.

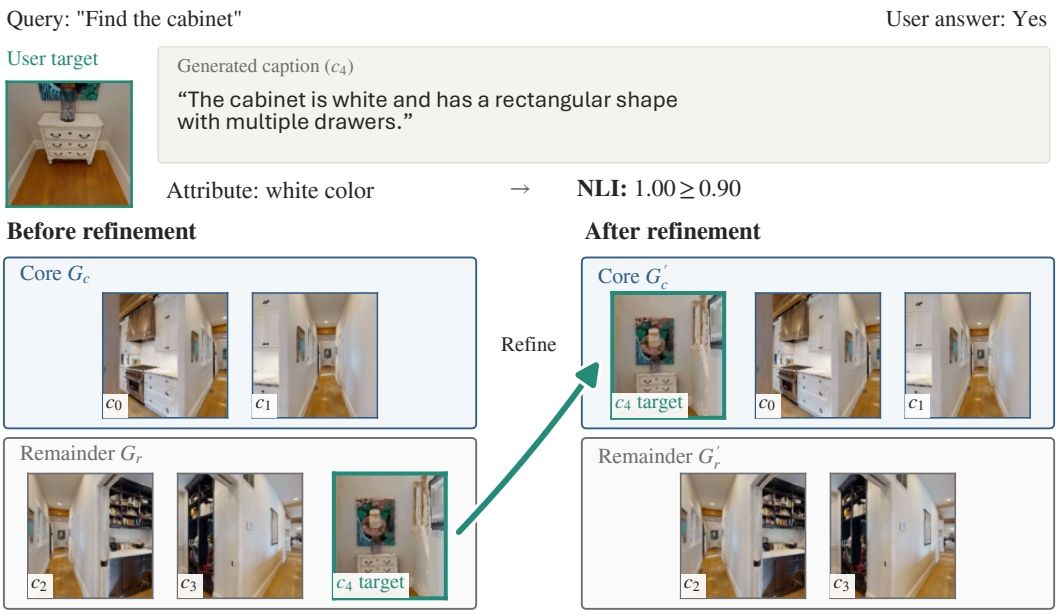
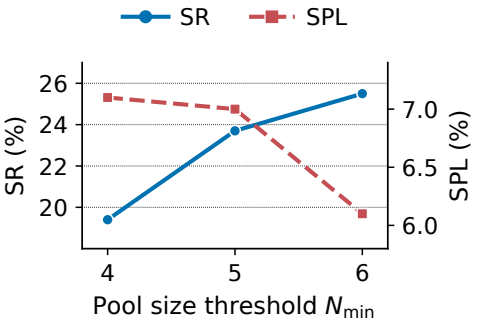


Figure 9: Attribute-guided group refinement (episode 1617). The quotation is the generated caption excerpt for c_4. Its support for "white color" exceeds the NLI threshold ($1.00 \geq \tau = 0.9$), moving the target from G_r into G_c'. For the same recorded Yes answer, the unrefined partition would prune c_4, while the refined partition retains it. Candidate images are crops of recorded views, with a zoom for c_4; the user target reference is unavailable to the NLI verifier.



$N_{\min}$	SR↑	SPL↑	RL↓	NQ↓	Steps↓
4	19.4	7.1	3.8	2.0	212.7
5†	23.7	7.0	4.2	2.2	257.0
6	25.5	6.1	4.5	2.4	299.3

†Default setting.

Figure 10: Effect of the candidate pool size threshold $N_{\min}$ that triggers the Recursive Comparison Stage on CoIN-Bench. (Left) SR and SPL under different $N_{\min}$ values. A larger $N_{\min}$ improves comparative judgment and thus increases SR, but leads to lower SPL. (Right) Detailed results for all evaluation metrics, including user burden and navigation cost. Based on this trade-off, we choose $N_{\min} = 5$ as the default setting on CoIN-Bench.

K Effect of the Pool Size Threshold

In Fig. 10, we further analyze the effect of the candidate pool size threshold $N_{\min}$ that triggers the Recursive Comparison Stage on CoIN-Bench. When $N_{\min} = 4$, the Recursive Comparison Stage is invoked earlier, yielding the highest SPL and fewest steps, but the candidate pool may not yet be sufficiently representative, resulting in a low SR of 19.4. Increasing $N_{\min}$ to 5 substantially improves SR to 23.7 (+4.3) while incurring only a modest SPL drop, suggesting that the additional exploration translates directly into SR gains by providing a more reliable comparison set. In contrast, $N_{\min} = 6$ provides only a limited SR gain (+1.8) but noticeably worsens efficiency and interaction cost. We therefore choose $N_{\min} = 5$ as the default setting on CoIN-Bench, as it offers the best balance between success rate, navigation efficiency, and user burden.

Table 12: Path efficiency across target distances on Val Seen. Distance is the geodesic distance from the agent's initial position to the target. No episode has a distance of 3 m or less. The three distance bins contain 25, 251, and 555 episodes, respectively. Results are obtained from separate evaluation runs using the same evaluation protocol as the main experiments. Lower mean steps are better.

Method	3-5 m SR	5-7 m SR	>7 m SR	Overall SR	Mean steps
AIUTA*	8.0	12.8	10.8	11.3	261.8
Pooled	28.0	22.3	13.3	16.5	**242.3**
ProCompNav	**28.0**	**31.1**	**20.2**	**23.7**	257.0

K.1 Navigation efficiency across target distance

Table 12 shows that ProCompNav outperforms AIUTA* across all geodesic-distance groups with comparable mean steps. Pooled Independent Matching accounts for much of this gain, while comparative judgment adds larger improvements at longer distances.

Figure 11 shows the three highest- and three lowest-SPL successes in a visualization rerun. Delayed candidate collection can prolong exploration and produce detours. In (f), only two candidates have been collected by step 400, below $N_{\min} = 5$. RCJ is then triggered by the step threshold, and the agent subsequently reaches the target.

L Stability Across Random Seeds

For each method, we use master seeds 0, 1, and 2 to initialize Python's `random`, NumPy, PyTorch, and CUDA random-number states. We also set `config.habitat.seed` to the master seed and enable episode shuffling, so the episode order varies across seeds.

Table 13: Evaluation with three seeds on Val Seen Synonyms. Values are mean ± standard deviation.

Method	SR (%)	SPL (%)
AIUTA*	15.1 ± 0.7	8.3 ± 0.3
Pooled	20.3 ± 2.7	6.5 ± 1.0
ProCompNav	**27.5 ± 1.5**	**8.5 ± 0.4**

Across three seeds, ProCompNav achieves the highest mean SR and SPL. Its SR margin over Pooled Independent Matching is larger than either method's standard deviation (Table 13).

M Results with Different MLLMs

Table 14: Results with different MLLMs on CoIN-Bench Val Seen Synonyms. SR is Success Rate.

Method	Qwen3-VL-8B	Gemma-3-12B
3D-Mem*	18.9	12.0
ProCompNav	**28.1**	**20.9**

ProCompNav outperforms 3D-Mem* with both Qwen3-VL-8B and Gemma-3-12B [23], showing that its advantage is not limited to one MLLM (Table 14). We evaluate both methods on the 359 Val Seen Synonyms episodes with the same MLLM in each comparison. Because 3D-Mem reports no CoIN-Bench result, 3D-Mem* is our reproduction.

High-SPL trajectories

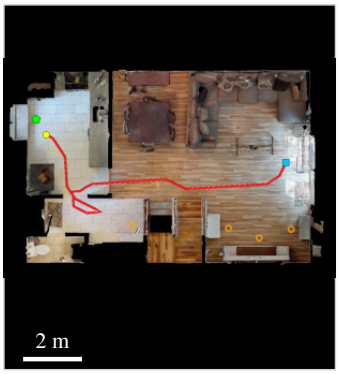
(a) Cabinet
Episode 1325



(b) Cabinet
Episode 977



(c) Cabinet
Episode 488



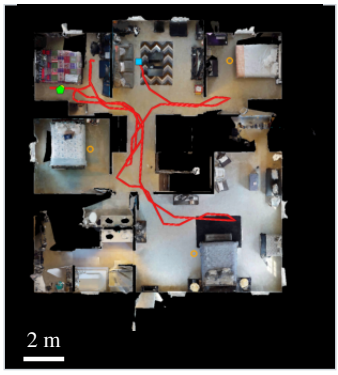
Metric	(a)	(b)	(c)
SPL	**88.4%**	**72.6%**	**63.9%**
Start-target distance	6.40 m	6.34 m	8.14 m
Traveled distance	7.24 m	8.74 m	12.73 m
Navigation steps	61	100	103
Collected candidates	6	5	5

Low-SPL trajectories

(d) Table
Episode 1019



(e) Bed
Episode 148



(f) Bathroom cabinet
Episode 1753



Metric	(d)	(e)	(f)
SPL	**0.7%**	**6.8%**	**10.4%**
Start-target distance	0.20 m	3.62 m	4.49 m
Traveled distance	27.87 m	53.11 m	43.22 m
Navigation steps	243	411	442
Collected candidates	6	6	2

Path Start Stop Target Distractor

Figure 11: Successful navigation trajectories from a visualization rerun of 12 preselected CoIN-Bench Val Seen episodes. The top and bottom rows show the three highest- and lowest-SPL successes, with metrics aligned below each row. Red lines show recorded routes. Start-target distance is the geodesic distance from the agent's initial position to the nearest target goal viewpoint used in SPL evaluation. Traveled distance sums recorded movement during candidate collection and final navigation. Collected candidates are counted after merging at the first RCJ call. Maps are independently scaled, with metric scale bars.

Table 15: Effect of the user simulator on ProCompNav performance on CoIN-Bench. We compare the original Qwen3-VL-8B user simulator with Gemini 3 Flash Preview.

User Simulator	Val Seen				Val Seen Synonyms				Val Unseen			
	SR ↑	SPL ↑	RL ↓	NQ ↓	SR ↑	SPL ↑	RL ↓	NQ ↓	SR ↑	SPL ↑	RL ↓	NQ ↓
Qwen3-VL-8B	23.7	7.0	4.2	2.2	28.1	8.5	4.3	2.2	17.0	6.2	4.2	2.3
Gemini 3 Flash Preview	23.1	6.7	4.1	2.1	29.0	8.5	4.1	2.1	15.5	5.3	4.0	2.1

Table 16: Per-episode computational cost comparison between AIUTA* and ProCompNav on the Val Seen Synonyms split (359 episodes). Although ProCompNav takes more navigation steps to proactively collect candidates, it reduces MLLM/LLM inference time by 63.8% relative to AIUTA*, replacing repeated open-ended MLLM scoring with lightweight NLI verification, and thus achieves a lower total wall-clock time.

Metric	AIUTA*	ProCompNav
Total runtime (s)	707.74	549.60
Number of steps	203.3	271.5
MLLM/LLM inference time (s)	229.01	82.89
NLI inference time (s)	-	1.29
Exploration time (s)	478.72	465.42

N Robustness Across User Simulators

We also evaluate ProCompNav using Gemini 3 Flash Preview as an alternative user simulator (Table 15). The results across SR, SPL, RL, and NQ are similar overall to those with Qwen3-VL-8B.

O Computational Cost

Table 16 breaks down the per-episode computational cost. All computational costs were measured on two NVIDIA RTX 3090 GPUs with 24GB memory each. The dominant saving comes from MLLM/LLM inference: AIUTA* invokes the MLLM repeatedly to score each candidate independently and to generate open-ended questions, accumulating 229.01s per episode, whereas ProCompNav concentrates its MLLM calls in the Pool Construction Stage's captioning and a small number of attribute extractions in the Recursive Comparison Stage, totaling only 82.89s. The NLI verifier used for DA scoring adds a negligible 1.29s.

P Instance Merge Analysis

The Pool Construction Stage attempts to consolidate observations of the same physical instance from multiple views into a single candidate c_i; we assess how reliably this consolidation succeeds for the target instance. For each episode in CoIN-Bench Val Seen, we feed the multi-view collage I_i of every candidate to a high-performance VLM oracle (Gemini 3 Flash Preview[1]) and label each candidate as containing the target or not. These VLM judgments can miss the target even in successful episodes, so target-presence counts can differ from the failure analysis. Among the 379 episodes whose pool is judged to contain the target (Fig. 12), 50.9% have the target's views consolidated into a single candidate (*Target merged*). In the remaining 49.1% the target's views are split across two or more candidates (*Target scattered*).

This incomplete consolidation harms downstream behavior in two ways. First, the multi-view description d_i is generated from a partial set of views per candidate, so the description used by the Recursive Comparison Stage misses information that other views of the same instance carry. Second, when the same target instance is treated as multiple distinct candidates, the Recursive Comparison Stage poses unnecessary discriminative questions to the user. Improving cross-view consolidation in the Pool Construction Stage is therefore a direct lever for both caption quality and user efficiency.

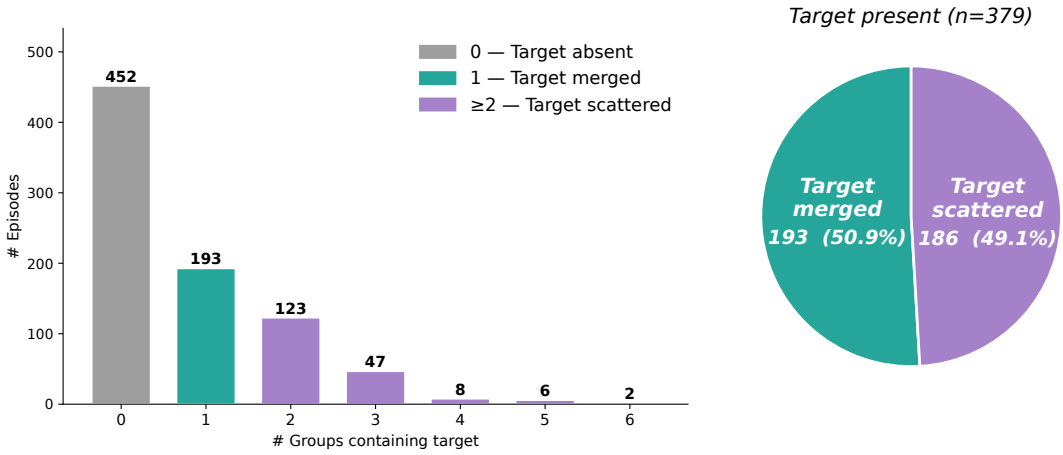


Figure 12: Instance merge quality of the Pool Construction stage on the CoIN-Bench Val Seen split, assessed using VLM judgments of candidate collages. **Left**: distribution of the number of candidates whose multi-view collage contains the target. *Target absent* (gray) marks episodes where no candidate contains the target. *Target merged* (teal) marks episodes where the target's views are fully consolidated into a single candidate. *Target scattered* (purple) marks episodes where the target's views are split across two or more candidates. **Right**: among the 379 episodes whose pool is judged to contain the target, the shares with the target merged and scattered.

Table 17: Success rate for different levels of target and distractor similarity on CoIN-Bench Val Seen. Results are obtained from separate evaluation runs using the same evaluation protocol as the main experiments.

Similarity	Ours	AIUTA*	Pooled	N
LOW	**22.9**	9.4	14.4	534
MED	**18.5**	10.3	13.0	146
HIGH	**31.8**	18.5	27.2	151

Q Performance Across Target-Distractor Similarity

We use Gemini 3 Flash Preview to group episodes by target-distractor similarity in color, texture, size, shape, and visible context. LOW, MED, and HIGH denote increasing similarity across these cues. ProCompNav outperforms both baselines across all three groups, including HIGH (Table 17).

R Human Evaluation Details

Both studies were conducted online using a within-subjects design. We pre-specified a target sample size of 20 participants per study and analyzed the first 20 completed sessions. The studies use target images and questions drawn from method logs. Questions were translated into Korean using Gemini 3 Flash Preview and manually checked before deployment.

Recruitment and consent. Participants volunteered in response to recruitment announcements shared with colleagues and acquaintances. All were Korean nationals in their twenties or thirties and participated from South Korea. The samples may overlap across the two studies. Before starting, participants read a Korean-language explanation of the study purpose and procedure and provided electronic consent by clicking "Agree and start." The specific hypotheses were not disclosed to reduce response bias. Incomplete sessions were excluded from the analysis.

Duration and compensation. Recruitment announcements estimated approximately 10 minutes per study. Among the analyzed participants, the median elapsed time from consent to final completion was 7.5 minutes in Study 1 and 7.4 minutes in Study 2, including the tutorial, waiting, responses, and

preference selection. No compensation was provided. In Study 2, participants were informed that waiting would occur and that they could do other work while waiting for a notification.

Collected information. Study records contain randomly generated participant IDs, responses, response times, preference choices, and experiment metadata such as conditions, questions, and event timestamps. The exported records contain no names, email addresses, ages, genders, or IP addresses. Only aggregate participant results are reported in the paper. Appendix R.1 provides English translations of the introductory instructions shown in the apps.

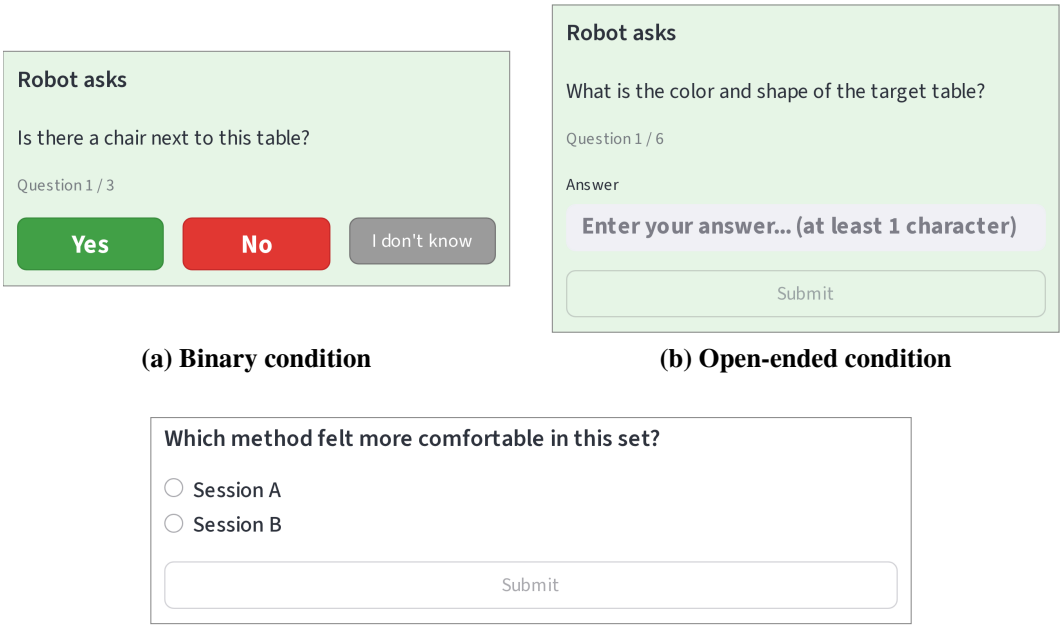
Study 1: question conditions. The binary-question condition presents yes/no/unknown buttons, whereas the open-ended-question condition presents a free-text field. Binary questions are drawn from ProCompNav logs, and open-ended questions from Pooled Independent Matching logs. Each participant completes five paired trials covering cabinet, bed, table, desk, and chair. All participants experience the same five target episodes, but the episode order varies across participants. Both conditions use batched delivery and the same target image within each pair. Whether binary or open-ended questions come first is balanced across participants to reduce order effects from familiarization or fatigue. Each participant answers 13 binary and 27 open-ended questions. We retain each method's original logged question sequences rather than equalizing question counts, preserving the question burden of each interaction strategy. Open-ended responses require at least one character, with no time, length, or keyword constraints. Figure 13 shows the interface used to answer both question types and report preference after each paired trial.

Study 2: question delivery. Three binary questions are presented in each condition without requiring navigation. We sample waiting times independently and uniformly within ranges to approximate unpredictable question timing during robot interaction and reduce anticipation based on elapsed time. Participants see expected ranges rather than a countdown. The ranges are informed by Val Seen log medians, with exploration delays halved to limit study duration and LLM processing delays retained. The interleaved schedule is adjusted to keep total waiting time comparable across conditions.

The batched condition uses an initial wait of 157-173 s and between-question intervals of 4.75-5.25 s. The interleaved condition uses an initial wait and between-question intervals of 57-63 s. The two target images are balanced across the batched and interleaved conditions across participants. Condition order is counterbalanced, and participants select their preferred delivery after experiencing both.

Measurement. Response time runs from question display to submission. Descriptive response times exclude outliers separately within each condition using the 1.5-IQR rule. Participant-level comparisons use each participant's median response time across all responses in each condition. In Study 1, every participant was faster in the binary condition. In Study 2, 12 of 20 participants were faster with batched delivery.

Preference and response length. In Study 1, participant preference is the majority of five paired choices. The binary condition was preferred by all 20 participants and in 96 of 100 paired trials. In Study 2, 14 of 20 participants selected batched delivery after experiencing both conditions. Open-ended responses are segmented by whitespace: 481 of 540 (89.1%) contain at most three words, with medians of one word and three characters. These are whitespace-delimited Korean words, not MLLM tokens, and are not directly comparable to the token-level Response Length reported for the user simulator.



(a) Binary condition **(b) Open-ended condition**

(c) Condition preference

Figure 13: Study 1 interface, translated into English and cropped, with text sizes adjusted to highlight the response controls. (a) The binary condition provides yes/no/unknown options. (b) The open-ended condition provides a free-text field. (c) Participants select their preferred condition after each paired trial. The study was conducted in Korean, and condition order was counterbalanced.

R.1 Participant Instructions

The following are English translations of the Korean introductory instructions shown before the "Agree and start" button. Icons and visual formatting are omitted. The recruitment announcements estimated 10 minutes for each study, while the Study 2 app estimated 8 minutes.

The accompanying code package includes the complete original Korean instructions, including tutorial and session-transition messages, in the study apps' `config.py` and `app.py` files under `user_study/`.

Study 1: introductory instructions.

> This experiment compares two ways in which a robot asks you questions. The two ways are presented in separate sessions, and you will complete five pairs of sessions.
>
> Purpose: This experiment investigates which of two ways for a robot to ask users questions feels more psychologically comfortable.
>
> Important notes: You may participate on either a PC or a mobile device. Please answer the robot's questions as conscientiously as possible. If you close the window, you cannot resume where you left off. Please complete the experiment in one sitting if possible.
>
> Estimated time: Approximately 10 minutes.
>
> Procedure: You will complete five sets. Each set concerns a different target object and consists of Session A, Session B, and a choice of which way was better. The target object image is always displayed at the top of the session. Depending on the session, you will answer by clicking a button (yes/no/don't know) or entering text. The experiment ends after all five sets.
>
> Data: Only an anonymous ID and your responses are stored. Personally identifying information is not collected.

Study 2: introductory instructions.

> This experiment compares two ways in which a robot asks you questions. Each way is presented in a separate session.
>
> Purpose: This experiment investigates which of two ways for a robot to ask users questions feels more psychologically comfortable.
>
> Important notes: Please participate on a PC only. Mobile devices are not supported. Headphones or earphones are recommended so that you can hear notifications. You do not have to answer the robot's questions perfectly. Please answer comfortably, and incorrect answers are acceptable. If you close the window, you cannot resume where you left off. Please complete the experiment in one sitting if possible.
>
> Waiting: You will spend some time waiting for the robot to explore within the page. Feel free to do other work while waiting. When the robot is ready to ask a question, a notification sound will play.
>
> Estimated time: Approximately 8 minutes.
>
> Procedure: You will complete two sessions. In each session, the robot briefly explores a room and then asks three yes/no questions about the target object. The robot asks questions to find the target object. You may do other work while it explores. Check the question when you hear the notification. The target object image is always displayed at the top of the session. Look at the image and answer each question with yes, no, or don't know. After the two sessions, complete a short survey about the two ways, and the experiment ends.
>
> Data: Only an anonymous ID and your responses are stored. Personally identifying information is not collected. Your responses are not graded for correctness.

Response and preference instructions. In Study 1, the practice screens instruct participants to "Answer by pressing one of the three buttons below" or "Enter an answer of at least one character in the field below, then submit." The main-trial instructions specify the sequence "Session A: answer several questions about a target object. Session B: answer in a different way about the same target object. Choose which way was more comfortable." The preference question is "Which way was more comfortable in this set?" with options "Session A way" and "Session B way." In Study 2, the tutorial explains that exploration in the main study takes approximately 57 seconds to 2 minutes 55 seconds depending on the session, but that the tutorial wait is shortened to 5 seconds. The preference screen states "You have experienced both ways. Please choose one" and asks "Which of the two ways you just experienced do you prefer?" Before choosing Session A or Session B, participants see each session described as either "a long wait followed by several consecutive questions" or "a short wait followed by several questions at regular intervals."

S Failure Analysis Details

We examine where navigation fails and why RCJ may prune the target. Figure 4 summarizes the failure stages, and Table 7 focuses on 134 target-pruning episodes. We use VLM judgments of candidate images to check whether the candidate finally selected as the target contains the intended instance. If it does but the episode fails, we count a navigation failure. These VLM checks are used only for analysis.

For each pruning episode, we give Gemini 3 Flash Preview[1] the target image and ask whether it shows the attribute selected in the pruning round. In a separate call, we provide the target candidate's multi-view caption and ask whether it states or clearly implies the same attribute. We also inspect the target's group before and after refinement and the Qwen3-VL-8B response. Each episode is assigned to the first applicable category below.

1. **Captions.** If Gemini judges that the caption describes the selected attribute but the target image does not show it, we assign **Caption Hallucination**. The reverse is **Caption Omission**. When both types occur in one episode ($n = 1$), we count it under hallucination.
2. **Core set.** If Gemini finds the selected attribute in neither the target image nor its caption, but the target is in the core set, we assign **Core-set False Positive**. A **No** answer from the user simulator then removes the core set, including the target.

3. **Refinement.** For a target in the remainder set, we assign **Refinement False Negative** if Gemini finds the selected attribute in both the target image and its caption, but refinement does not move the target to the core set. We assign **Refinement False Positive** if Gemini finds it in neither, but refinement moves the target to the core set.
4. **User simulator.** We assign **Simulator-Judge Mismatch** when Gemini's answer from the target image would retain the target, but the user-simulating MLLM Qwen3-VL-8B gives the opposite answer and the target is pruned.

Although Gemini is a high-performing MLLM, it may also misread the target image or interpret the question differently from Qwen3-VL-8B. Their disagreement alone therefore does not mean Qwen3-VL-8B is wrong. When judging the target image, Gemini is instructed to answer **No** if the selected attribute is absent or cannot be assessed from that image. The multi-view caption uses the agent's collected images. An attribute may be present on the target but not visible from the captured viewpoints, leaving it absent from the caption.

T Open-ended Feedback Corruption Example

Color — Blanket Episode 1729 · p=10%

Question What is the color and pattern of the target blanket, and does it have any visible logos or unique designs?

Original response

The target blanket is **pink** with a floral pattern featuring white flowers with green centers. It does not have any visible logos or unique designs beyond the floral motif.

Corrupted response

The target blanket is **blue** with a floral pattern featuring white flowers with green centers. It does not have any visible logos or unique designs beyond the floral motif.

Target image

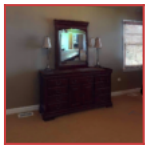


Corruption rule: replace the queried attribute value; keep other response content unchanged.

Figure 14: An actual open-ended Flip example from AIUTA* Val Seen logs (episode 1729, $p = 10\%$). The color value changes from pink (teal highlight) to blue (orange highlight), while all other response content remains unchanged. The target image is shown at right.

U Dialogue Examples

(a) Cabinet Query: "Find the cabinet" Episode 977

Target	AIUTA*	Pooled IM	ProCompNav
			
User simulator reference	Success	Failure	Success

AIUTA* 1 question

Q1 What is the color or finish of the target cabinet's surface?

A1 Based on the image provided, the target cabinet (a wooden dresser with a mirror on top) has a light brown wood finish. This finish appears to be polished or varnished, giving the wood a warm, slightly reflective sheen. The grain of the wood is visible and enhances the natural look of the furniture. The color is a medium to light brown, typical of many hardwoods like oak or maple that have been stained and finished for furniture. The surface is smooth and glossy, which is evident from how light reflects off it, particularly on the top of the dresser and the front of the drawers.

Pooled IM 5 questions

Q1 What is the color and material of the target cabinet?

A1 Based on the image provided, here is a description of the cabinet: - Color: The cabinet has a warm, medium-to-dark brown finish. The wood grain is visible, and the finish appears to be polished or varnished, giving it a slight sheen. - Material: The cabinet is made of wood. This is evident from the visible grain patterns on its surface and the classic wooden construction of its drawers and frame. The cabinet is a traditional-style dresser with multiple drawers and a mirror mounted on top. It is positioned against a dark gray wall next to a bed.

Q2 What color is the wall directly behind the target cabinet, and what is the finish and wood grain visibility of the cabinet itself?

A2 Based on the image provided: - The wall directly behind the wooden cabinet is painted a dark gray color. - The cabinet itself has a warm, medium-brown wood finish with a noticeable glossy or semi-gloss sheen, which reflects some light. The wood grain is visible, appearing as subtle, natural lines and patterns within the finish, especially on the drawer fronts and sides. The cabinet is a traditional-style dresser with multiple drawers and a mirror mounted on top.

Q3 What color is the wall behind the target cabinet, and is it dark gray or a different shade like dark blue-gray or slate blue-gray?

A3 Based on the image provided, the wall behind the wooden dresser (cabinet) is a very dark, muted tone. While it appears to be a shade of gray, it is not a pure, flat gray. The color has subtle undertones that suggest it is more accurately described as dark blue-gray or slate blue-gray. The lighting and shadows in the room make it difficult to be certain, but the hue leans toward a cool, bluish-gray rather than a neutral gray. It's a deep, sophisticated shade that complements the warm wood tones of the furniture. Therefore, the wall is best described as dark blue-gray or slate blue-gray, not just dark gray.

Q4 What is the color of the wood finish on the target cabinet, and is the wood grain visible?

A4 Based on the provided image, here is an analysis of the wood finish on the cabinet: - Color of the Wood Finish: The cabinet has a warm, medium-to-dark brown finish. It appears to be a stained wood with a noticeable sheen or gloss, which reflects some light and gives it a polished look. - Visibility of Wood Grain: Yes, the wood grain is clearly visible. The grain patterns are distinct and run horizontally across the drawer fronts and the cabinet body. The finish enhances the natural texture of the wood rather than hiding it. In summary, the cabinet features a warm, glossy brown stain that highlights its prominent wood grain.

Q5 Does the cabinet have visible wood grain and a glossy polished finish, or is it matte and painted?

A5 Based on the visual evidence in the image, the cabinet has a visible wood grain and a glossy polished finish. Here's why: * Visible Wood Grain: The surface of the dresser clearly shows the natural patterns and lines characteristic of wood. You can see the grain running along the panels of the drawers and the top surface, which is a strong indicator that it is made of real wood. * Glossy Polished Finish: The surface reflects light, creating bright highlights, especially on the top and front edges of the drawers. This sheen is characteristic of a polished or varnished finish, not a matte one. A matte finish would be flat and non-reflective. Therefore, the cabinet is made of wood with a glossy polished finish.

ProCompNav 2 questions

Q1 Property: light brown color	**A1** Yes
Q2 Property: rectangular shape with arched mirror top	**A2** Yes

Figure 15: Full dialogue comparison for episode 977. AIUTA* and ProCompNav succeed, whereas Pooled selects a distractor. All recorded user-simulator questions and answers are shown; ProCompNav questions are displayed as their recorded properties. The target image is the user simulator's reference. Pooled and ProCompNav views are reconstructed at logged poses; AIUTA* uses an approximate action replay.

(b) Desk Query: "Find the desk" Episode 534

Target	AIUTA*	Pooled IM	ProCompNav
			
User simulator reference	Failure	Failure	Success

AIUTA* 1 question

Q1 What is the color and material of the target desk's surface?

A1 Based on the image provided, here is a description of the target desk's surface: - Color: The surface is a light to medium brown, characteristic of natural wood. It has a warm, slightly weathered tone with variations in shade across the planks. - Material: The surface is made of wood. It appears to be constructed from multiple wooden planks or panels joined together, as evidenced by the visible seams and grain patterns. The finish is not glossy or varnished; it looks like a natural, possibly aged or distressed wood finish. In summary, the desk's surface is wooden and has a light to medium brown color.

Pooled IM 4 questions

Q1 What color is the frame of the photo on the target desk?

A1 Based on the image provided, the frame of the photo hanging on the wall above and to the left of the desk is black. It appears as a simple, thin, dark-colored frame that contrasts with the lighter wall and the photo within it.

Q2 What is the color of the frame of the picture on the target desk?

A2 Based on the image provided, there is a picture hanging on the wall above and to the left of the desk. The frame of this picture is white. It appears as a simple, clean border around the artwork, contrasting with the dark gray wall behind it.

Q3 What is the color of the frame of the picture located on the desk, near the front edge and arranged in a row?

A3 Based on the image provided, there is no picture frame located on the desk near the front edge that is arranged in a row. The only picture visible in the image is hanging on the wall to the left of the desk, and it has a dark, possibly black or dark brown, frame. Therefore, there is no picture frame on the desk that matches your description.

Q4 What is the color of the frame of the picture hanging on the wall above and to the right of the desk?

A4 Based on the image provided, there is a picture hanging on the wall above and to the left of the desk (from the viewer's perspective). The frame of this picture is white. It appears to be a simple, thin white frame surrounding a black-and-white photograph or print. The wall behind it is painted a dark gray color, which makes the white frame stand out. Therefore, the color of the frame is white.

ProCompNav 3 questions

Q1 Property: has papers on top	**A1** No
Q2 Property: rectangular shape	**A2** Yes
Q3 Property: white bookshelf adjacent	**A3** No

Figure 16: Full dialogue comparison for episode 534. AIUTA* accepts a distractor after one question; Pooled asks four questions and selects an incorrect candidate, whereas ProCompNav succeeds with three binary questions. All recorded user-simulator questions and answers are shown; ProCompNav questions are displayed as their recorded properties. Pooled collects four candidates without the target; ProCompNav collects five candidates. The target image is the user simulator's reference. Method images are illustrative reconstructions from the episode-start pose (AIUTA*) and selected-group coordinates (Pooled and ProCompNav).

V Asset Sources and Licenses

Both navigation benchmarks use scenes from the Habitat-Matterport 3D Dataset (HM3D) [18]. Rendered target images are subject to the Matterport academic-use agreement. Table 18 records the principal upstream sources and their publicly stated licenses or access terms.

Table 18: Principal asset sources and licenses. Asset names link to their original releases or license documents. A code license is distinguished from the terms for dataset annotations and scene data.

Asset or release	License or access terms
HM3D v0.2 scenes	Matterport academic-use agreement
CoIN-Bench v1 annotations	Not specified in the dataset release
PSL/TextNav annotations	Not specified in the dataset release
CoIN/AIUTA implementation	CC BY-NC 4.0
VLFM implementation	MIT
Habitat-Sim 0.2.4	MIT
Qwen3-VL-8B-Instruct	Apache 2.0
DeBERTa-v3-large	MIT
DINOv2-large	Apache 2.0
all-MiniLM-L6-v2	Apache 2.0
GroundingDINO Swin-T	Apache 2.0
MobileSAM	Apache 2.0
BLIP2 ITM ViT-g	MIT
Gemma-3-12B	Gemma Terms of Use
Gemini 3 Flash Preview	Hosted API, subject to provider terms
Qwen3.6-Plus	Hosted API, subject to provider terms

W Prompts

We provide the prompt templates used in ProCompNav and Pooled Independent Matching. Red placeholders are replaced with the corresponding inputs.

W.1 ProCompNav Prompts

The multi-view description prompt is shared by CoIN-Bench and TextNav; the remaining prompts are grouped by setting below.

Multi-view Object Description:

```
You are given an image which shows a {category} from multiple viewpoints.
You need to describe the {category}, by combining the information from these
different viewpoints.
First, describe the {category} in detail, focusing on its appearance and
distinctive features (use only:  color, shape).
Then, describe the other objects (use only:  color, shape) close to the
{category}, and their spatial relationships (use only:  ['next to', 'on top of',
'under']) relative to the {category}.
Mention all clearly visible nearby objects around the {category}.
```

This prompt is used to generate a unified caption from multiple viewpoints of the same object instance.

W.1.1 CoIN-Bench

Shared Attribute Extraction:

```
You are given a list of [{category}] descriptions.
Task: Identify properties that are shared by ALL [{category}s] in the list.
- Properties should be based on visible appearance or nearby surrounding objects.
- If a surrounding object is distinctive, explicitly name it.
- Each property should be specific but concise.
- Each property must be no more than 10 words.
- Output only properties, one per line.

List of descriptions:
{descriptions}

DO NOT USE the following properties: [{used_properties}].
Each property must apply to every object in the list.
Extract at most 10 valid shared properties.

Format:
property1
property2
property3
```

where {descriptions} is replaced by the candidate descriptions and {category} by the target instance category c. The used-property list excludes attributes already queried.

Binary User Simulation:

```
image: <image>
Does this [{category}] have the following property?
PROPERTY: [{property}]
Answer strictly with 'yes' or 'no'. Do not say anything else.
```

The image is the target-instance image available only to the user simulator. The selected attribute-value pair replaces {property}.

W.1.2 Text-goal Navigation (TextNav)

Text-goal Attribute Extraction:

```
You are given a single [{category}] description.
Task: Extract distinctive properties of this [{category}].
- Properties must be specific to this [{category}], not generic.
- Include only directly observable physical attributes of the [{category}] itself
or objects adjacent to it.
- Exclude atmosphere, emotional tone, interpretive statements, and descriptive
effects.
- Each property should be specific but concise.
- Each property must be no more than 10 words.
- Output only properties, one per line.

Description:
{text_goal}

Extract at most 10 valid properties.

Format:
property1
property2
property3
```

This prompt extracts the fixed attribute set from the initial TextNav goal description.

Text-goal Matching:

```
You are given two descriptions of a [{category}].
Description A (reference): {reference_description}
Description B (candidate): {candidate_description}
Notes:
- Each description can include both the object description and nearby surrounding
context.
- Focus on whether they refer to the same object; surrounding context may differ.
Question: Do Description A and Description B refer to the same [{category}]?
Answer strictly with 'yes' or 'no'. Do not say anything else.
```

This prompt is used to check whether a final candidate matches the given detailed description in the TextNav setting.

W.2 Pooled Independent Matching Prompts

These prompts are used for Pooled Independent Matching on CoIN-Bench.

Target Fact Update:

```
You are an intelligent embodied agent tasked with finding a specific target
{category}.

You know the following facts about the target {category}:
<START_TARGET_PICTURE_FACTS> {target_facts} <END_TARGET_PICTURE_FACTS>

You recently detected another picture and asked to the human several question:
<START_OF_ORACLE_ANSWER>
{question_answer_pairs}
<END_OF_ORACLE_ANSWER>

Task: Update the target facts with this new information. Be concise. Do not
include information that are uncertain.

YAML_START
facts: <updated facts as a single text line>
YAML_END # must be present to get the information back
Provide your reasoning step-by-step, after the YAML_END tag.
```

This AIUTA prompt updates the accumulated facts after each user response. Each question-answer pair is inserted as `- Question: <question> - Answer: <answer>`.

Tie-breaking Candidate Selection:

```
collected target fact: {target_facts}

{candidate_list}

Pick the ONE candidate whose description best matches the target fact.
Respond with only the id. No other text.
```

When multiple candidates share the highest final score, {candidate_list} contains their descriptions, each prefixed with `candidate <id>:`, and the LLM selects one candidate.

Candidate Scoring and Question Generation: This single prompt, inherited from AIUTA [22], returns both a matching score and a question. During the first pass, its question is used to collect user feedback. During the second pass, its score is used to rank candidates against the final facts; the generated question is not asked.

```
You are an intelligent agent equipped with an RGB sensor, object detector, and
Visual Question Answering (VQA) model.
Your goal is to identify a target {category} based on a scene description and
prior knowledge of the target.

Scenario:
The object detector has identified a scene containing a {category}, and the VQA
model has provided the following description:

<START_OF_DESCRIPTION>
{candidate_description}
<END_OF_DESCRIPTION>

Target object information:
Based on previous interactions, you know the target picture has the following
characteristics:
<START_TARGET_PICTURE_FACTS>
{target_facts}
<END_TARGET_PICTURE_FACTS>

Task:
1.  Similarity analysis.
Analyze how closely the detected scene description aligns with the known facts
about the target {category}.  Provide a similarity score between 0 and 10, where:
- 0 = The detected {category} is not the target object.
- 10 = The detected {category} is definitely the target object.
- If no information about the target is available, the score should be -1.

2.  Question Generation:
- The question is for the target object, not the detected one.
- Ask exactly one specific, relevant, and human-answerable question related to
the target object that maximizes information gain for identifying the target
{category}.
- Do not ask speculative or irrelevant questions
- The question should be grounded in observable or known details from the scene,
focusing on key characteristics that can help confirm or refute the identity of
the target object.

Ensure your response follows the format below:
YAML_START # must be present to get the information back
similarity_score:  <similarity score>
questions:
  <question_number>:  <question_content>
YAML_END # must be present to get the information back

Provide your reasoning step-by-step for the similarity score and questions, after
the YAML_END tag.
```

Open-ended User Simulation: Following AIUTA [22], the generated question itself is passed to the MLLM together with the target-instance image, without an additional response instruction. We use Qwen3-VL-8B as described in Section 5.1.

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 and Introduction (Sec. 1) state the shift from independent matching to comparative judgment, the candidate-pool construction, and the reduction in user response length. These claims are realized by the method (Sec. 4) and supported by the experiments (Sec. 5).

   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: The Limitations paragraph in Sec. 7 discusses navigation evaluation in simulation, dependence on exploration and detection to include the target in the pool, failure to recover a target pruned by incorrect feedback, and the minimum candidate-pool-size requirement.

   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 training-free method and contains no formal theoretical results.

   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: Secs. 4 and 5.1 describe the algorithm, benchmarks, splits, metrics, models, trigger conditions, and evaluation budgets. Appendices B, C, D, and E provide pool construction, NLI scoring, TextNav adaptation, and exploration details, respectively. The refinement threshold is specified in Appendix J and Table 11, and the prompt templates are in Appendix W.

   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: [Yes]

   Justification: ProCompNav and Pooled Independent Matching are included in our public repository, https://github.com/ml-postech/ProCompNav, with environment, data preparation, and evaluation instructions. AIUTA is available from its original release. Some ablation-specific scripts are not included. The accompanying code package includes the human-study code and method-generated question stimuli with target images. Participant responses remain private. Appendix R reports the study protocol, analysis methods, and aggregate results.

   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. 5.1 reports the splits, models, exploration backbone, trigger conditions, budgets, and success criterion. The overlap threshold and $K = 6$ view clustering are specified in Appendix B, the NLI score conversion in Appendix C, the refinement setting $\tau = 0.9$ in Appendix J and Table 11, and the exploration heuristics in Appendix E.

   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: [Yes]

   Justification: Table 13 reports SR and SPL means and standard deviations across three seeds for the Val Seen Synonyms split. This repeated-seed evaluation is limited to that split. Appendix L specifies seeds 0, 1, and 2 and the controlled random-number states.

   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: [No]

   Justification: Appendix O specifies two NVIDIA RTX 3090 GPUs with 24 GB each for the runtime measurements in Table 16. Gemini 3 Flash Preview and Qwen3.6-Plus API calls are excluded from these runtime measurements.

   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: [No]

   Justification: We did not offer compensation because we treated the brief sessions as voluntary participation with limited burden. Prior IRB review was not sought due to time constraints during study preparation, and the applicability of institutional review requirements was not checked. Appendix R describes recruitment, electronic consent, compensation, procedures, and collected information.

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: Secs. 1 and 7 motivate reducing user interaction burden in instance navigation, but the paper does not include a discussion of potential negative societal impacts.

    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: We release no new high-risk pretrained model or scraped dataset. The method is built on existing public benchmarks and pretrained backbones.

    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: [No]

    Justification: Appendix V and Table 18 identify the principal asset sources, publicly stated licenses, and HM3D access terms. HM3D was obtained using an account that accepted the applicable agreement. The license applicable to the CoIN-Bench and TextNav episode annotations has not been verified from their original releases, so this part of the license review remains unresolved.

    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: [Yes]

    Justification: The public pipeline code provides environment, data preparation, and evaluation instructions and an MIT license for our code. The accompanying human-study package contains the apps, questions from ProCompNav and Pooled logs, rendered target images, preparation and analysis scripts, and a README explaining the studies and data fields. Participant responses are excluded, and third-party data terms are documented separately.

    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: [Yes]

    Justification: Appendix R describes the study procedures and compensation. Appendix R.1 provides translated instructions and identifies the study apps in the accompanying code package that contain the complete original Korean instruction text. Figure 13 shows the Study 1 interface.

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: [No]

    Justification: The user studies involved no anticipated risks. Participation was voluntary, and Appendix R provides detailed descriptions of the study procedures.

    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: Qwen3-VL-8B generates candidate descriptions and attribute candidates and serves as the default user simulator (Sec. 5.1). DeBERTa-v3-large verifies attribute support (Appendix C). Gemma-3-12B is used in the backbone comparison (Table 14). Qwen3.6-Plus is used to extract atomic hints (Appendix H). Gemini 3 Flash Preview is accessed via API for the alternative user simulator (Table 15), target-presence and similarity labeling (Appendices P and Q), post-hoc failure analysis (Appendix S), and Korean translation of human-study questions (Appendix R).

    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.