Politecnico di Torino

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program in Artificial Intelligence (38^{th} cycle)

Robust Embodied Navigation via Active Perception and Human-Agent Interaction

By

Francesco Taioli

******

Supervisor(s):
Prof. Marco Cristani, Supervisor
Prof. Alessandro Farinelli, Co-Supervisor

Politecnico di Torino
2026

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and do not compromise in any way the rights of third parties, including those relating to the security of personal data.

Francesco Taioli
2026

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

Abstract

Artificial intelligence is now pervasive in our daily lives, from conversational agents and creative content generation to protein structure prediction and autonomous vehicles. As AI systems become increasingly integrated into the physical world, embodied AI, where intelligence is coupled with sensors and actuators to interact with the environment, has emerged as a key frontier, often regarded as the next industrial revolution with significant societal and economic impact. Despite impressive results, embodied agents are typically trained and evaluated under ideal benchmark settings, assuming perfect perception, clearly specified goals, or error-free, unambiguous language instruction. Yet, such assumptions rarely hold in real-world settings, where out-of-distribution conditions may occur and humans often provide underspecified or erroneous input. This thesis addresses these challenges by enhancing the robustness of embodied agents through active perception and human–robot interaction.

*To address **sensor unreliability**, we consider the problem of Active Visual Search (AVS), where the agent is tasked to find a specific object in an environment. In this setting, small objects, motion blur, low resolution, partial views, and heavy occlusions severely degrade both detection quality and planning accuracy. To this end, we present **POMP-BE-PD** [116], an unsupervised Monte-Carlo POMDP planner that explicitly models the uncertainty inherent in object detection to optimize the agent behavior. Specifically, the agent maintains a probability distribution on a 2D floor map, and incorporates the awareness that an object detector may fail into the aforementioned probability.*

*For the **instruction robustness**, we consider the fact that human instructions may be: (i) inaccurate or even prone to error, due to bad memory or cognitive/perceptual impairments; and (ii) ambiguous, as users may not be able or willing to supply all details of the task in advance. To this end, in [118] we first propose a novel task and benchmark of Detection and Localization of Instruction Errors, in which*

*different types of errors are injected into textual instructions. Moreover, we propose Instruction Error Detection & Localizer (**IEDL**) [118], a method to detect and localize errors within a sub-sentence distance on the original instruction. We extend this method in **I2EDL** [117], enabling agents to query users for clarification while minimizing their cognitive burden.*

*Finally, to handle **instruction ambiguity**, we introduce the Collaborative Instance object Navigation (**CoIN**) task [119], a new task setting where the agent actively resolves uncertainties about the target instance during navigation in natural, template-free, open-ended dialogues with humans. Our proposed agent, Agent-user Interaction with UncerTainty Awareness (**AIUTA**), leverages large Vision-and-Language models to reason about uncertainty and engage in efficient human-agent interaction.*

Contents

List of Figures viii

List of Tables xiii

1 Introduction 1

1.1 Contributions . . . 3

1.2 Outline . . . 5

1.3 Publications . . . 6

2 Preliminaries 9

2.1 Embodied Tasks . . . 9

2.2 Embodied Agents . . . 10

2.3 Vision and Language . . . 12

3 Agent Robustness via Sensor Unreliability 14

3.1 Related Works . . . 14

3.2 POMP-BE-PD . . . 17

3.2.1 Method . . . 17

3.2.2 Experimental Results . . . 26

3.2.3 Conclusion . . . 34

4 Agent Robustness via Instruction Understanding 35

4.1 IEDL/I2EDL 36
4.1.1 Related Works 37
4.1.2 Task & Benchmark 40
4.1.3 Method 44
4.1.4 Experimental Results 46
4.1.5 I2EDL: Interactive IEDL 51
4.1.6 Conclusion 57
4.2 AIUTA 58
4.2.1 Related Works 59
4.2.2 Task & Benchmark 61
4.2.3 Method 64
4.2.4 Experimental Results 71
4.2.5 Conclusion 77

5 Conclusions **78**
5.1 Summary of the main contributions 78
5.2 Limitations & Future Work 79
5.3 Embodied AI: The Next Era 83
5.4 Potential Impact 84

References **87**

Appendix A Collaborative Instance Navigation **108**
A.1 AIUTA: Prompts 108
A.1.1 P_{init} - Initial Description 108
A.1.2 $P_{details}$ - Gather Additional Information 108
A.1.3 P_{check} - Check detection with LVML 109
A.1.4 $P_{self question}$ - Extract attributes and generate Self-Questions . 109

A.1.5 $P_{refined}$ - Refined image description 110
A.1.6 P_{score} - Alignment score 111
A.2 AIUTA: Algorithm 113
A.2.1 Self Questioner 114
A.2.2 Interaction Trigger 115

List of Figures

1.1 The *human-agent–environment* interaction. 2

3.1 An agent is initialized in a known environment with the task of visually searching for a target object, *i.e.*, to localize the object and approach it. (a) 3D reconstruction of the environment; the agent has to navigate toward the target (yellow star) through the possible shortest path (highlighted in green) while avoiding longer trajectories (in orange) without missing entirely the target (in red). (b) Corresponding 2D grid map of the scene in our POMCP modeling: blue dots are the possible object locations, purple crosses are the possible robot poses. 15

3.2 Overall architecture of our proposed method POMP-BE-PD. The red box represents prior knowledge pushed into the POMCP module, the grey box represents the exploration strategy to detect the target object, the yellow box represents the probabilistic docking strategy to reach the destination pose and the green box represents the probability distribution over the locations. Math notation: state s_t, action a_t, pose p_t, observation o_t, POMCP state sequence $s_{\{0..T_d\}}$, docking state sequence $s_{\{T_{d+1}..T\}}$, complete state sequence $s_{\{0..T\}}$. 18

3.3 The two cases considered when creating the vector D. Example derived from Home_003_2. In case (a), the objective is to determine the location of the object and assign probabilities in the form of a multivariate normal distribution. In (b), we assign low probabilities to the locations inside the FOV, and high probabilities to the locations outside it. Note: we assign different scales to the colorbar for ease of visualization. 24

3.4 Evolution of the probabilities p_j inside Home_016_1 using the proposed approach POMP-BE-PD. In step (a), we initialize the agent in the environment; we highlight the target position and a false positive area. From step (b) to (c), the robot explores the top area; in step (d) we show the robustness of our approach to a false positive; finally, in step (e), we identify highly probable locations, locating the target in step (f). 25

3.5 Corresponding 2D floor maps (not in scale) for the test scenes from AVBD of 3 different difficulty levels (as in [106]). For each environment, we report the name. As the difficulty increases, we can note an increment of possible object locations and more difficult spatial layouts. 26

3.6 We aggregated the episodes by the minimum number of steps to reach the object, thus incorporating the difficulty of the episode. In Figure (a) the results for the Easy scenarios; in Fig. (b) Hard Scenarios; in Fig. (c) the Medium ones and finally, in Fig. (d), the sum of all scenarios. Results using the object detector provided by [4], both during planning and docking. Focusing on the POMP-PD method (yellow bar), we can observe the increment of efficiency and efficacy due to the introduction of the Belief Update (green bar), since both methods do not change the exit condition during planning (Probabilistic Detection). 30

3.7 Percentage of error of POMP, POMP-PD and POMP-BE-PD, averaged over all scenarios. The errors are categorized into three types: Localization, Docking and Other. We used the object detector provided by [4], during both planning and docking. 32

4.1 An agent navigates in a scene, following instructions expressed in natural language, for example *"Exit the bathroom and go left (✓ right), then turn left at the big clock and go into the bedroom and wait next to the bed."* By just changing "right" to "left" in the instruction, the agent terminates the exploration in the wrong location, ignoring the fact that along the path it did not see the "big clock" (yellow arrow). 36

4.2 Comparison of the Success Rate (SR) of different methods (in order [5, 6, 43, 43, 57, 57]) working on continuous environments. We show the SR on the standard R2R-CE dataset split `Val Unseen` (green) and the drop in SR performance when errors are present (red). Interestingly, we see up to -25% drop in SR when up to three errors among *{Direction, Room, Object}* per episode are present. 37

4.3 Architecture of our proposed *IEDL* model, representing the scenario depicted in Fig 4.1. The frozen policy π follows Instruction Υ, producing a sequence of observations $\mathcal{O}$. Then, a panoramic encoder and a language encoder produce, respectively, the trajectory visual features Γ and instruction features Υ. We then feed the trajectory set Γ and Υ to a cross-modal multilayer transformer to produce visual-language aligned representations. Finally, two task-specific heads perform *Instruction Error Detection* and *Instruction Error Localization*. 44

4.4 Success Rate upper bound ($\overline{\text{SR}}$) at different step-t. 56

4.5 SR and SIN plotted at different step-t for localization threshold $\tau_l = 1$. Specifically, dashed lines indicate the value for the SIN metric, while solid lines indicate the SR. The *"Always Ask"* baseline always interacts with the user from step-t onwards. 57

4.6 Collaborative Instance object Navigation (CoIN) task illustration. A human provides a request (*"Find the picture"*) in *natural language*. The agent has to locate the object in a *completely unknown* environment *without any target image as input*, interacting with the user only when needed via *template-free*, *open-ended natural-language dialogues*. Our method, **A**gent-user **I**nteraction with **U**ncer**T**ainty **A**wareness (**AIUTA**), minimizes user interactions by equipping the agent with two modules: a **Self-Questioner** and an **Interaction Trigger**. The **Self-Questioner** leverages an LLM and VLM in a self-dialogue to describe the agent's observation and then extract additional relevant details, with a novel entropy-based technique to reduce **hallucinations and inaccuracies**, producing a refined **detection description**. The **Interaction Trigger** uses this refined description to decide whether to pose a question to the user (①,③,④), continue the navigation (②), or halt the exploration (⑤). 59

4.7 CoIN-Bench can be very challenging when only given the instance category to the agent. We highlight the target instance with red borders, while the distractor instances that exist in the same scene are marked with blue borders. 64

4.8 CoIN-Bench evaluation setup. (Left) Real human responding to the agent's question. (Right) Simulated user-agent interactions, where the user responses are provided by a VLM with access to a high-resolution target instance image for scalable and reproducible experimentation. 65

4.9 Graphical depiction of **AIUTA**: left shows its interaction cycle with the user, and right provides an exploded view of our method. ① The agent receives an initial instruction I: "Find a c =<object category>". ② At each timestep t, a zero-shot policy π [141], comprising a frozen object detection module [74], selects the optimal action a_t. ③ Upon detection, the agent performs the proposed AIUTA. Specifically, ④ the agent first obtains an initial scene description of observation O_t from a VLM. Then, a ***Self-Questioner*** module leverages an LLM to automatically generate attribute-specific questions to the VLM, acquiring more information and refining the scene description with reduced attribute-level uncertainty, producing $S_{refined}$. ⑤ The ***Interaction Trigger*** module then evaluates $S_{refined}$ against the "*facts*" related to the target, to determine whether to terminate the navigation (if the agent believes it has located the target object ⑥), or to pose *template-free, natural-language* questions to a human ⑦, updating the "*facts*" based on the response ⑧. 65

4.10 AIUTA generates questions covering a wide range of attributes, such as color, material, style, and spatial arrangement. 73

4.11 Examples from IDKVQA, showing images and the questions generated by the LLM. 76

4.12 τ sensitivity results. For each method, 30 new τ values are sampled symmetrically around the optimal threshold τ^*. The x-axis shows the set size as a percentage of the original IDKVQA dataset size, while the y-axis displays the normalized ER $\Phi_{c=1}$. 77

List of Tables

3.1 Results on three scenes from AVDB using GT objects annotations. All methods are compared using the protocol defined in [106]. The asterisk $^{(*)}$ indicates that the evaluation is performed on a different subset of objects. 28

3.2 Result of different versions of improved POMP with more scenes per difficulty level in AVD. POMP-BE is POMP with the improved Belief Update. Result using the ground truth annotations instead of the detector, using 2^{10} simulations during the planning phase. The new Belief Update consistently increase the efficiency of the exploration phase, thus reducing the Average Path length, and increasing the SR and SPL. 31

3.3 Results of POMP and variations of POMP-BE-PD with more scenes per difficulty level in AVDB [3] using the object detector provided by [4]. 33

4.1 Statistics of the R2RIE-CE benchmark. 44

4.2 Results of our proposed *IEDL* method on our proposed benchmark. We show the SR and SPL metrics of the frozen policy, and the drop in SR performance when errors are present (ΔSR %). We then analyze the classification (AUC) and Localization (ATD) performance of different methods. Error types with * indicate benchmark with common sense. AUC is highlighted as it is the main metric. 48

4.3 Results show the increase of SIN (in %) under different paradigms of interaction on R2RIE-CE benchmark, with localization threshold $\tau_l = 1$, weighting factor $\lambda = 0.01$ from step $p = 4$ onwards. The primary metric SIN is highlighted. Under the *"No Interaction"* column, we report the SR, SPL metrics of the BEVBert policy [5], also showing the Success Rate Upper Bound ($\overline{\text{SR}}$). For *I2EDL*, we set detection threshold $\tau_d = 0.6$. Error type based on R2RIE-CE *Val Unseen* Dataset. 54

4.4 CoIN-Bench statistics: Average (standard deviation) number of distractors, geodesic distance to the goal, and number of episodes per split. 63

4.5 CoIN-Bench is challenging. AIUTA, while being *training-free*, achieves strong performance by outperforming trained policies (top rows) and significantly surpassing the zero-shot VLFM, across *all* splits, through effective user interaction. In contrast, policies trained on GOAT-Bench (denoted with †), the foundation of CoIN-Bench, fail to generalize to novel categories (Val Unseen). We report the SR (main metric, in **bold** *w.r.t* training free-methods), SPL, and the number of questions NQ. Input types: *c* for object category, *d* for its description. 72

4.6 Real human *vs* simulated user-agent interaction. 74

4.7 Ablation of AIUTA components on the CoIN-Bench Train split. . . 74

4.8 Results of different selection functions and their corresponding *Effective Reliability* rate $\Phi_{c=1}$ [135] on the IDKVQA dataset. 75

Chapter 1

Introduction

Artificial Intelligence (AI) is now deeply woven into our everyday life, from chatbots and personal assistants [92, 37, 122], to autonomous driving [137], protein structure prediction [1], and medical diagnosis systems [128, 82]. These systems have demonstrated remarkable capabilities across a range of tasks traditionally reserved for human experts, thanks to the rapid progress in large-scale datasets and models, multimodal learning, and data-driven architectures. Yet, we are still in the early days of a new industrial revolution, where AI systems are increasingly transitioning from static, digital domains into the physical world. In this context, *embodied agents*, equipped with sensors and actuators, perceive, reason, and interact with their surrounding environment, becoming the foundation for future industrial automation, logistics, elderly care, and household personal assistance. *Embodied AI* refers to the study and development of such agents operating in physical or simulated environments. Unlike traditional AI systems that rely on static datasets, embodied agents must process sensory input (from both the environment and *humans*), make sequential decisions, and act in real time while continuously adapting to environment feedback, as illustrated in Fig. 1.1. This paradigm requires the integration of perception (*i.e.*, accurate understanding of the environment), language understanding (as natural language is the intuitive way of human-agent communication), planning (to determine not only the next action but also long-term strategies), and control (to translate high-level actions into low-level ones) in order to perform physically grounded tasks such as navigation, object manipulation, and interactive instruction following. The economic implications of this shift are expected to be substantial.

Fig. 1.1 The *human-agent–environment* interaction.

Emerging research forecasts the deployment of 63 million humanoid robots in the U.S. by 2050, equivalent to a 3$ trillion in salary replacement [84].

Despite the rapid progress of embodied AI, developing embodied agents remains highly challenging. First, one of the primary obstacles is the need for state-of-the-art simulators to train these agents. While real-world training would be ideal, it is largely infeasible due to several limiting factors: *(i)* it is slow, as agents require the equivalent of years of interaction experience to learn effectively; *(ii)* it is expensive, since diverse physical environments are needed; and *(iii)* it is not reproducible, as real-world conditions are inherently variable and difficult to control. Second, there is a lack of diverse, high-quality real-world datasets and tasks on which to train and evaluate these agents. Moreover, when agents trained in simulation are deployed in the real world, they often underperform, a phenomenon known as the sim-to-real gap. For example, physical dynamics and visual observations in simulation often lack the visual fidelity, richness and variability of real-world environments, limiting the agent's ability to generalize effectively. Robustness is therefore a core requirement for embodied agents. Unlike static AI models, they operate in noisy, dynamic and ambiguous environments, where sensor errors, instruction ambiguity, and unpredictable conditions are the norm, not the exception. Yet, current benchmarks largely overlook these challenges, training and evaluating agents under idealized conditions.

In this thesis, we identify two complementary sources of uncertainty that undermine agent robustness. The first originates internally, from the agent's perspective: sensors and learned models (*e.g.*, object detectors) are inherently imperfect and may fail under real-world conditions (*sensor unreliability*). The second originates externally, from the human user: natural language instructions are often ambiguous, incomplete, or erroneous (*instruction uncertainty*). Despite their importance, both

sources are largely overlooked in current benchmarks, which assume perfect perception and unambiguous instructions. Both sources of uncertainty must be addressed for embodied agents to operate reliably in uncontrolled environments. We now motivate each in turn.

Motivation for agent robustness via sensor unreliability. When deployed in a real home and asked to "*find a tea cup on the table*", an agent must deal with unexpected challenges such as occlusion, motion blur, low resolution, and partial views. These conditions degrade the performance of object detectors, models commonly used to localize household items. Yet, such challenges are rarely accounted for in simulation-based training. As a result, agents trained in simulation may fail to generalize when exposed to the noise and uncertainty of real-world sensory input, which can account for small to large, out-of-distribution objects.

Motivation for agent robustness via instruction understanding. Task ambiguity is also largely ignored in current benchmark settings. For instance, if a user requests "*find a tea cup on the table*", what should the agent do if there are multiple tea-cups on the table? What if there are several tables, each with a tea cup on it? Or what if the user implicitly refers to "their" specific tea cup, relying on shared context the agent doesn't possess? Now consider navigation instructions such as "*Exit the bathroom and go left, then turn left at the big clock and go into the bedroom. Wait next to the bed for further instructions*". In existing benchmarks, such instructions are typically correct, complete, and specific. However, this is rarely the case in real-world scenarios. People often give ambiguous or incomplete directions, based on imperfect memory or assumptions about shared knowledge. Moreover, subjects with cognitive or perceptual impairments may also provide inaccurate instructions, for example, referencing landmarks that don't exist or using wrong spatial relations [44].

1.1 Contributions

This thesis contributes to the development of robust embodied agents by addressing challenges in sensor unreliability and imperfect human instructions. Accordingly, we organize our contributions into two main lines of work:

Contributions on robustness to sensors' unreliability.

- **POMP-BE-PD [116] (Section 3.2):** We introduce POMP-BE-PD, an unsupervised method for active visual object search (`InstanceObjectNav`) in environments where only a 2D floor map is available. The approach enhances robustness to sensor failures by explicitly modeling object detector unreliability within the agent's planning framework. Specifically: *(i)* the agent requires no training phase; *(ii)* it maintains a probability distribution over the 2D map throughout exploration; and *(iii)* the awareness that an object detector may fail is incorporated into the probability distribution by exploiting the object detector's statistics.
  We extensively evaluate our method, following the AVDB benchmark, achieving state-of-the-art results. Moreover, several ablation studies demonstrate the strength of our POMP-BE-PD. On average over all the environments, we increase the success rate by a significant 35% while decreasing the average path length by 4% with respect to the previous formulation.
  Project page at intelligolabs.github.io/unsupervised_active_visual_search.

Contributions on instruction robustness.

- **IEDL [118] and I2EDL [117] (Section 4.1):** For the first time, we introduce a novel benchmark dataset that injects various types of instruction errors into the `VLN` task (using Habitat-sim [105]), simulating common human mistakes (Section 4.1.2). We show that state-of-the-art agents suffer significant performance drops on this benchmark, highlighting a lack of robustness to imperfect input. We then propose the *Detection and Localization of Instruction Errors* task and *Instruction Error Detection & Localizer* (IEDL), a novel Instruction-Trajectory compatibility transformer model that aligns natural language instructions with agent observations to detect inconsistencies. Our proposed method, composed of a detection and localization heads, can detect and localize errors within a sub-sentence distance on the original instruction, showing better performance than baselines. As a further experiment, we show how a pre-trained IEDL can be used as a semi-automatic tool for potentially identifying error-containing episodes, isolating 8 episodes with ground-truth annotation errors in the validation set.

  In follow-up work, we introduce *Interactive Instruction Error Detection & Localizer* (I2EDL [117], Section 4.1.5), a model that detects instruction er-

rors and selectively triggers user-agent interaction. We also propose a new metric that captures both navigation performance and the cost of interaction. Compared to baselines, we show that our proposed I2EDL is generally more effective in improving navigation performance when erroneous instructions are given, while lowering the interaction load.
Project page at intelligolabs.github.io/R2RIE-CE.

- **AIUTA [119] (Section 4.2):** We introduce, for the first time, the *Collaborative Instance object Navigation* (CoIN) task, a new `InstanceObjectNav` setting where agents resolve ambiguity about the target object instance through *natural, open-ended, template-free* dialogues with humans during navigation. To support this task, we release CoIN-Bench, a curated benchmark featuring challenging multi-instance scenarios. By simulating users with a Vision-Language model (VLM), we show that it is possible to create a reproducible environment for studying and researching human–agent collaboration. Additionally, we propose *Agent-user Interaction with UncerTainty Awareness* (AIUTA), a novel training-free method that leverages VLM and Large-Language models (LLMs) for human-agent interaction reasoning. Through extensive experiments (using Habitat-sim [105]), we show that existing trained methods fail to generalize to unseen categories, while our training-free AIUTA, using a novel self-dialogue mechanism and uncertainty estimation, achieves strong performance across all validation splits. Moreover, our simulated user-agent interaction is in line with human evaluation, enabling scalable and reproducible experiments.
Project page at intelligolabs.github.io/CoIN.

1.2 Outline

The remainder of this thesis is organized as follows, progressing from foundational concepts to our two main contributions on agent robustness.

Chapter 2 introduces the preliminaries and foundational concepts used throughout this thesis. We begin by defining the core tasks of embodied AI addressed in this work, namely Instance Object Navigation (`InstanceObjectNav`) and Vision-and-Language Navigation (`VLN`). Next, we introduce the MDP/POMDP framework, a strong mathematical formalism that allows us to model these tasks rigorously. Finally, we provide background on vision-and-language alignment using Deep Learning ap-

proaches, reviewing models and techniques that enable agents to ground language in perception. These foundations provide the necessary background for understanding the methods and formulations used in the subsequent chapters.

Building on these foundations, **Chapter 3** addresses agent robustness in the presence of *sensor unreliability*. In particular, we introduce POMP-BE-PD, an unsupervised POMDP agent which incorporates the awareness of potential object detector failures into the planning procedure.

While Chapter 3 focuses on internal uncertainty (sensor unreliability), **Chapter 4** addresses the external source: *instruction uncertainty*. Specifically, in Section 4.1, we study the impact of erroneous language instructions by introducing new benchmarks that simulate human-like mistakes, along with a model capable of detecting and localizing these errors within the instruction. Then, in Section 4.2, we focus on ambiguous instructions, *i.e.*, situations where users cannot or do not specify task details fully. We introduce a benchmark for such under-specified inputs and propose a model for uncertainty-aware, interactive human–agent reasoning.

Finally, **Chapter 5** summarizes the main findings, discusses limitations, and outlines open challenges and future directions for robust embodied AI. We also reflect on the broader impact of this work for the computer vision and robotics communities.

1.3 Publications

The following section lists the author's publications. Articles included in this thesis are shown in bold, while others are not part of the manuscript.

1. Francesco Taioli, Francesco Giuliari, Yiming Wang, Riccardo Berra, Alberto Castellini, Alessio Del Bue, Alessandro Farinelli, Marco Cristani, Francesco Setti. (2024). [116]
   Unsupervised Active Visual Search With Monte Carlo Planning Under Uncertain Detections.
   In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

2. Francesco Taioli, Edoardo Zorzi, Gianni Franchi, Alberto Castellini, Alessandro Farinelli, Marco Cristani, and Yiming Wang. (2025). [119]

Collaborative Instance Object Navigation: Leveraging Uncertainty-Awareness to Minimize Human-Agent Dialogues.
In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

3. Francesco Taioli, Stefano Rosa, Alberto Castellini, Lorenzo Natale, Alessio Del Bue, Alessandro Farinelli, Marco Cristani, Yiming Wang. (2024). [118]
Mind the Error! Detection and Localization of Instruction Errors in Vision-and-Language Navigation.
In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

4. Francesco Taioli, Stefano Rosa, Alberto Castellini, Lorenzo Natale, Alessio Del Bue, Alessandro Farinelli, Marco Cristani, Yiming Wang. (2024). [117]
I2EDL: Interactive Instruction Error Detection and Localization.
In *IEEE International Conference on Robot and Human Interactive Communication (ROMAN)*.

5. Francesco Taioli, Federico Cunico, Federico Girella, Riccardo Bologna, Alessandro Farinelli, and Marco Cristani. (2023). [115]
Language-Enhanced RNR-Map: Querying Renderable Neural Radiance Field maps with natural language.
In *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*.

6. Andrea Toaiari, Federico Cunico, Francesco Taioli, Ariel Caputo, Gloria Menegaz, Andrea Giachetti, Giovanni Maria Farinella, and Marco Cristani. (2023). [126]
SCENE-pathy: Capturing the Visual Selective Attention of People Towards Scene Elements.
In *International Conference on Image Analysis and Processing (ICIAP)*.

7. Luigi Capogrosso, Federico Girella, Francesco Taioli, Michele Chiara, Muhammad Aqeel, Franco Fummi, Francesco Setti, and Marco Cristani. (2024). [15]
Diffusion-Based Image Generation for In-Distribution Data Augmentation in Surface Defect Detection.
In *Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications.*

8. Lia Morra, Alberto Azzari, Letizia Bergamasco, Marco Braga, Luigi Capogrosso, Federico Delrio, Giuseppe Di Giacomo, Simone Eiraudo, Giorgia Ghione, Rocco Giudice, Alkis Koudounas, Luca Piano, Daniele Rege Cambrin, Matteo Risso, Marco Rondina, Alessandro Sebastien Russo, Marco Russo, Francesco Taioli, Lorenzo Vaiani, and Chiara Vercellino. (2023). [85] Designing Logic Tensor Networks for Visual Sudoku puzzle classification. In *CEUR Workshop Proceedings*.

Chapter 2

Preliminaries

2.1 Embodied Tasks

Embodied AI has progressed rapidly, evolving from constrained, low-dimensional inputs to increasingly general-purpose agents that can reason, interact, and understand multiple modalities. Early work primarily focused on simple 3D navigation tasks such as PointGoal Navigation (`PointGoalNav`), where agents are initialized at a random position within a 3D environment, and tasked to reach a target position relative to the agent position [105], without using any maps. In this task, agents are typically equipped with an RGB-D, GPS, and compass sensors (*i.e.*, they have access to their location coordinates and relative position to the target goal). Notably, with the development of Deep Reinforcement learning, distributed training and high-throughput simulator (Habitat [105]), not only did [136] reach state-of-the-art results and essentially solved the `PointGoalNav` task, but showed that the navigation policy and scene understanding learned during the task can be transferred to other embodied tasks, serving as a foundation.

As the field progressed, more realistic and complex tasks were introduced. One such task is the Object-Goal navigation (`ObjectNav`), where agents must navigate unseen 3D environments to locate *any* instance of a specified object category (*e.g.*, *"Find a plant"* or *"Find a sofa"*). Being a challenging task, requiring spatial, semantic, and object-level reasoning, `ObjectNav` was simplified by restricting the set of object categories (*i.e.*, to 6 in the HM3D [97] scene dataset and 21 in the MP3D [18]). However, `ObjectNav` is only loosely aligned with real-world needs, where humans often

seek a *specific* instance of an object, not just any example of a category. To address this, Ammirato et al. [3] introduced the Active Vision Dataset Benchmark (AVDB), focusing on the Instance Object-Goal task (`InstanceObjectNav`), which requires agents to find a particular instance of a target object. Specifically, AVDB contains real indoor environment scans (in the form of RGB-D images) with 33 instances of annotated real objects. On the following line of research, OVON [142] extended `ObjectNav` to open-vocabulary settings, allowing agents to search for objects described by categories not seen during training. Other research followed in the same direction. For example, Krantz et al. [58] proposed the `InstanceImageNav` task, where the target object is specified via an image. Most recently, GOAT-Bench [55] unified multiple task formulations: category-based [142], detailed language-based (`InstanceObjectNav`) or image-based (`InstanceImageNav`), in an open vocabulary fashion. As language is the most intuitive way to interact with an agent, a parallel line of research focuses on the Vision-and-Language Navigation [57] task (`VLN`), with the challenging goal of autonomously guiding the agents toward a target position by following a series of instructions, expressed in natural language, *e.g.*, *"Exit the kitchen, go to the door on your left end enter the bedroom. Then stop."*

2.2 Embodied Agents

An embodied agent aims to learn a policy π, *i.e.*, a probability distribution over actions given environment states. In the context of embodied navigation, the agent operates in complex, partially structured environments and must make decisions based on visual, spatial, and semantic inputs (*e.g.*, RGB images, depth, lidar, instruction goal, etc). The navigation tasks introduced in Section 2.1, namely `ObjectNav`, `InstanceObjectNav`, and `VLN`, can be naturally framed as Markov Decision Processes (MDP) [12], a foundational model for solving sequential decision-making problems in reinforcement learning.

Formally, an MDP is defined by a tuple $\langle S, A, R, T, \gamma \rangle$, where:

1. S is a finite set of *observable* states;
2. A is a finite set of actions;

3. $R : S \times A \to \mathbb{R}$ is the reward function, which assigns a scalar reward to each state-action pair, representing the immediate reward gained after taking action a in state s;

4. $T : S \times A \to \Pi(S)$ is the *state-transition model*, mapping each state-action pair to a probability distribution over next world states;

5. $\gamma \in [0,1)$ is the discount factor.

In practice, the assumption that the agent has access to the full environment state is often unrealistic. Instead, agents typically rely on partial sensory observations that provide incomplete and *noisy* information. To address this, the MDP framework is often extended to Partially Observable Markov Decision Processes (POMDPs) [50], which is the standard formalism for modeling sequential decision processes under uncertainty. A POMDP is a tuple $\langle S, A, R, T, O, Z, \gamma \rangle$, where

1. S, A, T, γ and R describe an MDP;

2. Z is a finite set of observations the agent can experience, and;

3. $O : S \times A \to \Pi(Z)$ is the observation model; mapping each state-action pair to a probability distribution over possible observation.

The partial observability of the environment state is modeled by maintaining, at each time step t, a probability distribution over all possible states, referred to as the *belief* B. With this formulation, the agent no longer conditions its policy on the true state, but instead on its belief. POMPD solvers compute, in an exact or approximate way, a policy function $\pi : B \to A$, which maps beliefs to actions. The agent's objective is to maximize the expected sum of discounted rewards,

$$E[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)],$$

by selecting the optimal action a_t at each time step t, given the current state s_t. The discount factor γ reduces the importance of future rewards and ensures convergence of the sum.

2.3 Vision and Language

Aligning vision and language representations (*e.g.*, the semantic meaning of text with images) is a long-standing problem that has been extensively studied over the past decade, evolving through several distinct phases. Early work focused on image captioning, where models learned to generate textual descriptions from images. A notable work is the model by Vinyals et al. [130], which combines Convolutional Neural Network (CNN [63]) for image encoding and Long Short-Term Memory (LSTM) [41] for sequence generation. The development of transformers [129] marked a significant shift toward unified vision–language representations. Models such as ViLBERT [77] and LXMERT [120] introduced a cross-modal attention mechanism, allowing the model to attend jointly to language and visual features, producing a cross-modality representation. While these models improved generalization across multiple tasks, they were typically trained on curated datasets and required task-specific fine-tuning for each application.

A major breakthrough came with CLIP [96], which demonstrated that large-scale pre-training on image-caption pairs (*i.e.*, 400 million pairs) enables models to learn state-of-the-art vision-language representation from scratch. Specifically, CLIP consists of an image encoder and a language encoder trained using contrastive learning to align their embeddings in a shared semantic space. Notably, CLIP supports zero-shot transfer to a wide variety of downstream tasks, often achieving competitive or even state-of-the-art performance without task-specific supervision. Its embedding have since become a foundation for many vision-and-language applications, including embodied AI [54, 115, 118, 117], image generation [99], anomaly detection [66] and interpretation of medical images [125].

More recently, Vision-Language Models (VLMs), such as BLIP-2 [65], LLaVA [68, 69] and Qwen-2.5VL [9] have revolutionized the vision-language grounding tasks, such as visual-question answering, visual recognition, document parsing, object detection and even video understanding. These models achieve stronger multimodal alignment, scale more effectively with data and model size [52], and often require less task-specific tuning than earlier approaches [134]. VLMs extend the capabilities of Large Language Models (LLMs) by integrating visual inputs into the language generation process. A typical VLM architecture consists of three main components:

1. A vision encoder (often CLIP or CLIP-like model, based on the ViT architecture [27]), that converts an input image into a sequence of visual tokens.
2. A multimodal projector (often a two-layer multi-layer perceptron) that projects the visual embedding into the same embedding space as the language model.
3. A pre-trained language model, which generates output text conditioned on the visual and textual input (*i.e.*, prompt).

More formally, we consider an auto-regressive VLM, where:

1. $\mathbf{X}_I$ denotes the image representation (*i.e.*, image tokens)
2. $\mathbf{X}_P$ represents the prompt text tokens, and
3. $\mathbf{X}_H$ refers to the history tokens generated at previous time steps (the model is auto-regressive)

At each time step t, the VLM computes a conditional probability distribution p over the vocabulary $\mathbf{y} \in \mathbb{R}^w$, expressed as:

$$p_{\text{VLM}}(\mathbf{y_t} \mid \mathbf{X}_I, \mathbf{X}_P, \mathbf{X}_H).$$

While large-scale pre-training equips Vision–Language Models (VLMs) with general multimodal understanding, post-training is crucial for aligning these models with task-specific behavior and user intent. One of the most effective strategies is instruction tuning, where VLMs are fine-tuned on curated datasets containing pairs of inputs (*e.g.*, images and prompts) and desired outputs (*e.g.*, responses or captions) that reflect human-preferred behavior [68, 92]. Together, vision-language models, LLMs and VLMs have become central to advances in modern embodied AI systems. In this thesis, these models enable agents to ground natural language instructions and detect inconsistencies given an instruction (Section 4.1), and to perceive the environments in detail while engaging in human-agent dialogues (Section 4.2).

Chapter 3

Agent Robustness via Sensor Unreliability

Robustness to sensor unreliability is a fundamental requirement for embodied agents operating in real-world environments. Indeed, in practice, the quality of visual observations is highly unpredictable, affected by objects in the far field, low-resolution cameras, motion blur, occlusion, and partial views. These issues often lead to unreliable detections, including bot false positives and missed detections. During exploration, such errors could lead to agents terminating the exploration prematurely at an incorrect location, significantly reducing task success rates.

In this chapter, we address this challenge by proposing a sensor-aware agent for Active Visual Search (AVS). Specifically, AVS requires an agent to locate a specific target object in an unfamiliar environment, with only a 2D floor map as prior knowledge, as shown in Fig. 3.1. Our proposed method, POMP-BE-PD, enhances robustness to sensor failures by explicitly modeling object detector unreliability within the agent's planning framework.

3.1 Related Works

Tehre are two main research topics related to this work, which will be briefly surveyed in the following.

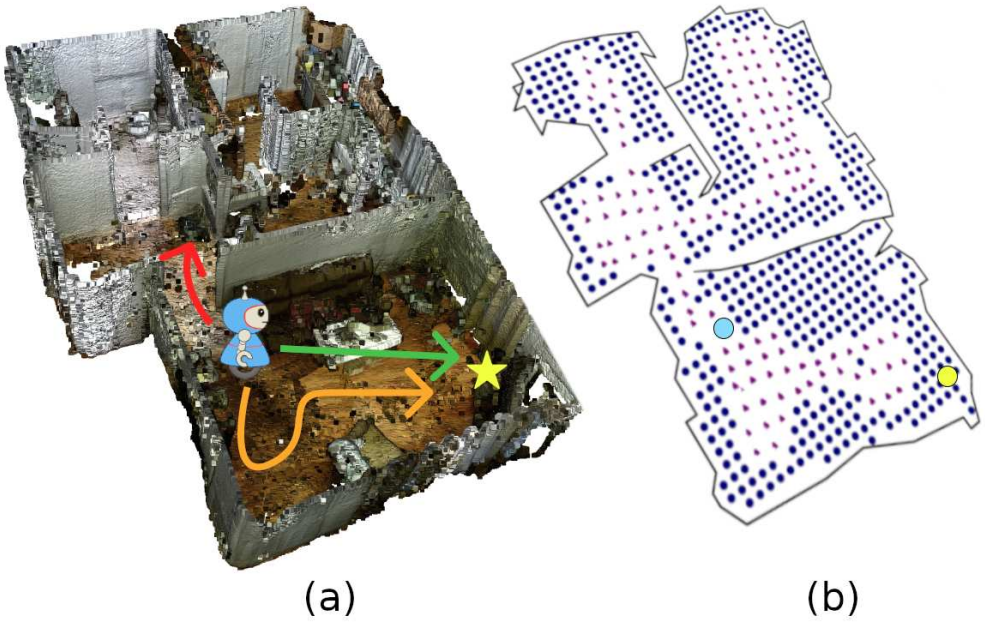


Fig. 3.1 An agent is initialized in a known environment with the task of visually searching for a target object, *i.e.*, to localize the object and approach it. (a) 3D reconstruction of the environment; the agent has to navigate toward the target (yellow star) through the possible shortest path (highlighted in green) while avoiding longer trajectories (in orange) without missing entirely the target (in red). (b) Corresponding 2D grid map of the scene in our POMCP modeling: blue dots are the possible object locations, purple crosses are the possible robot poses.

Active Visual Search. Early approaches to Active Visual Search (AVS) explored online reinforcement learning methods, where agents used current RGB-D observations and pose information to plan online their next actions [132, 35]. Other methods leveraged spatial relationships between objects to constrain the search space. For instance, intermediate objects like tables were used to infer likely locations for target objects such as chairs. Kunze et al. [62] proposed a probabilistic framework where the presence of co-occurring objects increases the likelihood of the target's presence, offering a soft constraint to guide search behavior. With the rise of deep learning, AVS has increasingly adopted Deep Reinforcement Learning (DRL) methods [106, 140, 40], where visual embeddings guide the agent's policy. For example, Han et al. [40] introduced a Deep Q-Network (DQN) that takes CNN-based RGB features and bounding box detections as input. However, their model assumes the target is initially detectable. To address this limitation, Schmid et al. [106] proposed EAT, a model that uses feature embeddings from both the current

scene and a target candidate crop proposal to guide action selection. Similarly, GAPLE [140] incorporates depth information alongside RGB features to enhance policy learning. Despite its generalization claims, GAPLE's performance depends heavily on extensive training with synthetic environments, such as the House3D simulator and the SUNCG dataset. This reliance on large-scale simulation is a common drawback among DRL-based methods, including those that utilize A3C algorithms [83], LSTM-based memory architectures [86], or transformers integrated with deep Q-learning [29]. Some works have proposed explicitly disentangling visual perception (*e.g.*, attention on relevant regions) from navigation [98], thereby improving task success rates. Others have introduced graph-based models, where spatial relationships among objects are encoded via Graph Convolutional Networks to guide navigation policy [104]. External commonsense knowledge has also shown advantages for object localization via spatial graph [36]. In contrast to the works discussed above, our agent performs efficient online planning without requiring any training, explicitly incorporating scene knowledge into the planning process.

Monte Carlo Planning. Partially Observable Markov Decision Processes (POMDPs) provide a foundational framework to model sequential decision-making in environments where full observability is not guaranteed [50]. Since computing exact solutions for non-trivial POMDPs is generally intractable, one of the most effective approximation strategies is Monte Carlo Tree Search (MCTS) [124, 22, 14], which is well-suited for large, uncertain domains due to its scalability and online nature. A seminal application of MCTS to POMDPs is the Partially Observable Monte Carlo Planning (POMCP) algorithm [110], which combines particle filtering for belief representation, online Q-value estimation via MCTS, and an efficient belief update mechanism. POMCP has inspired multiple extensions: BA-POMCP [53] introduces adaptive learning of environment dynamics; Amato and Oliehoek [2] extend the approach to multi-agent settings; and Lee et al. [64] address scenarios with cost constraints. Further improvements incorporate domain knowledge or symbolic reasoning into the planning pipeline, *e.g.*, leveraging known variable dependencies [16], enforcing safety guarantees [17], or generating interpretable policies via Satisfiability Modulo Theory (SMT) [80]. Earlier studies also explored probabilistic motion planning under uncertainty. For instance, [113] addressed planning in static environments, while [31] incorporated perception uncertainty and incompleteness into motion planning frameworks. In contrast to prior work, the proposed method special-

izes POMCP specifically for Active Visual Search (AVS). While prior approaches such as POMP [132] and POMP++ [35] applied POMCP to AVS, they exhibit key differences: POMP relies on standard belief updates and assumes perfect detection, whereas POMP++ targets fully unknown environments. Our method, instead, focuses on sensor-aware planning in semi-known settings, explicitly modeling detector unreliability and improving robustness without requiring any offline training.

3.2 POMP-BE-PD

In this Section, we present our proposed agent, POMP-BE-PD, which enables robust navigation by incorporating object detector failure awareness directly into the planning process. Specifically, we substitute deterministic detections with a probabilistic model, using Bayesian inference to maintain a distribution over all possible object locations. Building on this, we introduce an improved belief update strategy that increases planning efficiency in large state spaces, resulting in shorter overall path lengths. Finally, we enhance the agent with a novel docking procedure that improves reliability by leveraging the scene information accumulated during exploration.

Section Organization. We begin by describing the method in Section 3.2.1. We then present experimental results and analysis in Section 3.2.2, followed by a conclusion in Section 3.2.3.

3.2.1 Method

We consider a scenario in which an agent operates in a known environment, where the only available prior information is a 2D floor map. The agent's goal is to search for and approach a specific object. To do so, it must actively explore the space, detect the target, estimate its location on the map, and navigate toward it. The agent's pose at time step t is represented as $p_t = \{x_t, y_t, \theta_t\}$ where x and y are the 2D spatial coordinates and θ denotes the agent's orientation. We depict the overall architecture in Fig. 3.2.

At each step, the agent selects an action a_t from a fixed action set $A =: \{$`Forward, Backward, Turn Left, Turn Right` $\}$. Rotations are executed with a fixed an-

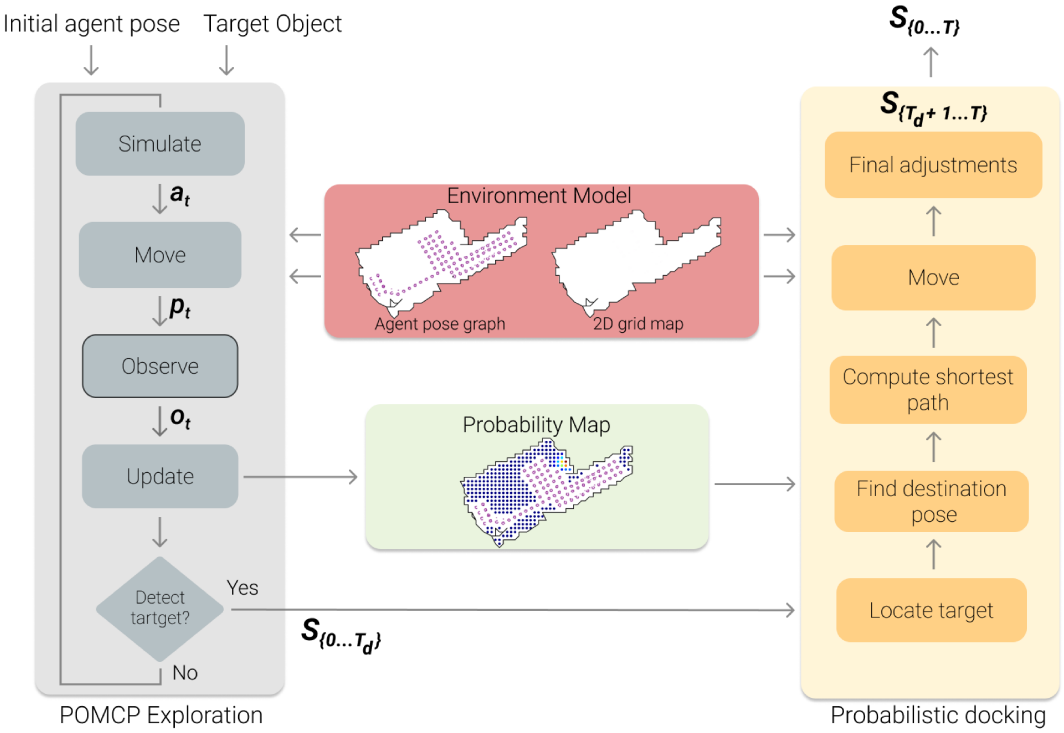


Fig. 3.2 Overall architecture of our proposed method POMP-BE-PD. The red box represents prior knowledge pushed into the POMCP module, the grey box represents the exploration strategy to detect the target object, the yellow box represents the probabilistic docking strategy to reach the destination pose and the green box represents the probability distribution over the locations. Math notation: state s_t, action a_t, pose p_t, observation o_t, POMCP state sequence $s_{\{0..T_d\}}$, docking state sequence $s_{\{T_{d+1}..T\}}$, complete state sequence $s_{\{0..T\}}$.

gular increment. Upon reaching a new pose p_t, the agent receives an observation by applying an object detector to the image captured via an onboard RGB-D camera. The environment is modeled as a 2D grid map (Fig. 3.1 (b)), where each cell is categorized as follows:

- *"Visual occlusion"*: occupied by obstacles (*e.g.*, walls or furniture) that block the agent's line of sight;
- *"Empty"*: traversable by the agent but cannot host the target object;
- *"Candidate"*: a potential location for the target object, neither occluded nor traversable.

3.2.1.1 Partially Observable Markov Decision Processes

We model the Active Visual Search (AVS) task as a Partially Observable Markov Decision Process (POMDP). A POMDP is defined by the tuple $(S, A, O, T, Z, R, \gamma)$, where:

- S is the set of partially observable states,
- A is the set of possible actions,
- Z is the set of possible observations,
- T: $S \times A \to \Pi(S)$ is the *state transition model*,
- O: $S \times A \to \Pi(Z)$ is the *observation model*,
- R: $S \times A \to \mathbb{R}$ is the reward function, and
- $\gamma \in [0, 1)$ is the discount factor.

The agent's objective is to maximize the expected sum of discounted rewards,

$$E[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)],$$

by selecting the optimal action a_t at each time step t, given the current state s_t. The discount factor γ reduces the importance of future rewards and ensures convergence of the sum. Due to partial observability, the agent maintains a belief distribution B over the state space, representing its uncertainty about the true state. POMDP solvers aim to compute either exact or approximate solutions in the form of a *policy*, defined as a function π: $B \to A$, which maps belief states to actions.

3.2.1.2 Partially Observable Monte Carlo Planning

POMCP [110] is an online solver for POMDPs that leverages Monte Carlo Tree Search (MCTS) to approximate the optimal action at each decision step. Starting from the agent's current belief, represented as an unweighted particle filter, POMCP performs multiple simulations to construct a search tree and estimate action Q-values. The agent then executes the action associated with the highest estimated value. One

of the key advantages of POMCP is its scalability to large state spaces. This is achieved by generating only the portion of the policy relevant to the belief states encountered during execution, rather than computing a full policy. Moreover, the local policy approximation is computed online through simulations using a generative model $\mathcal{M}(s,a)$ that produces the next state and corresponding observation, based on the environment's transition and observation models. In the following, we summarize the main phases of the POMCP algorithm.

Particle Initialization. The search tree begins with a root node representing an empty history h (no actions or observations yet). The belief at the root is initialized using a particle filter, where each particle corresponds to a randomly sampled hidden state (*e.g.*, a target object's position), drawn from a uniform distribution over all possible states.

Simulations and Statistics Update. For each time step t, POMCP performs a set number of simulations starting from the current history h. A particle representing a state s, is randomly sampled from the particle filter of node h, representing the agent's belief. A simulation trajectory then unfolds from s by selecting actions and generating new states and observations using the model $\mathcal{M}(s,a)$. Actions are chosen using the UCT algorithm if the current history lies within the tree, or using a uniform random policy otherwise. After all the simulation steps are performed, the total accumulated reward is used to update statistics (*i.e.*, visit counts and Q-values) for all nodes traversed in the simulation passing through h.

Action Selection in the Environment. Once all simulation steps are complete, the agent selects the action at the node h that has the highest estimated Q-value. This action is then executed in the real environment.

Belief Update. After executing the selected action and receiving the resulting observation o, the agent moves to the next node $h' = hao$ in the tree. The belief is updated by transferring particles from h to h', and the rest of the tree is pruned.

Particle Reinvigoration. If the new node h' does not contain enough particles (*e.g.*, due to insufficient overlap with particles from h), new particles are generated. This

is done via local perturbations of existing particles and a rejection sampling strategy that ensures consistency with the belief in h'. These new particles must represent states that are reachable from the previous belief given the action a and observation o.

3.2.1.3 Exploration, Localization and Approach.

Components. Let n denote the number of distinct poses the agent can assume, m the number of objects present, and k the number of candidate positions where each object might be located. We can now describe each component in detail:

(i) The first component is a *pose graph* $\mathcal{G}$, where each node represents a possible agent pose (out of the n), and edges define valid transitions, *i.e.*, an edge exists if a pose can be reached from another via a single action. Graph $\mathcal{G}$ enforces the movement constraints of the environment, restricting the agent from performing infeasible transitions.

(ii) We then define the set $\mathcal{H} = \{1, \ldots, k\}$, which enumerates all potential object locations. Each index in $\mathcal{H}$ refers to a specific location in the environment topology where the target object may be placed.

(iii) The third component is the *observability matrix* $\mathbf{L} = (l_{i,j}) \in 0, 1^{n \times k}$, where $l_{i,j} = 1$ indicates that the object location j is visible from agent pose i. This visibility is computed using a function f_L based on the environment's geometry, agent field of view, and line-of-sight constraints, applied to the pose graph $\mathcal{G}$ and candidate locations $\mathcal{H}$. The matrix $\mathbf{L}$ is used in the observation model used in *simulation*: an observation of 1 is returned if the agent is at pose $\hat{i} \in \mathcal{G}$ and the target is located at $\hat{j} \in \mathcal{H}$ such that $l_{\hat{i},\hat{j}} = 1$.

Notably, in simulation, the position of the target is known (defined in the sampled particle at the beginning of the simulation), making it possible to compute observations deterministically. Conversely, in the real world, the observations rely on an object detector.

Both in simulation and in real execution, a positive reward is given when the object is observed, and a negative reward is applied otherwise, accounting for energy or time spent on movement. To discourage repetitive exploration, the agent

maintains a memory of previously visited poses. Revisiting a pose results in a high negative penalty. At every real-world step, the agent receives a binary observation from the detector: 1 if the object is detected, and 0 otherwise. The agent's belief is represented as an approximate probability distribution over all candidate object locations, encoding uncertainty in the target's position and forming the hidden state in POMCP. If the agent fails to detect the object after a predefined number of steps, the search ends and a failure is reported.

Belief Update. In the previous POMP [132] formulation, belief updates are performed using the standard POMCP strategy [110]. However, this approach presents two main issues due to the large cardinality of the state space in AVS. Specifically, the state space includes both the agent's pose and the target object's location. Since the object can be located in any candidate position, and the number of simulations is inherently limited, some valid states may not be sampled during the simulation phase. Consequently, these states can only be recovered during the reinvigoration step. If not recovered, they are effectively removed from the belief and cannot be reintroduced, even if they are valid positions. A second issue arises from the reinvigoration mechanism itself. In standard POMCP, new particles are drawn from the previous belief distribution. This can induce a feedback loop: particles that survive a belief update are more likely to be selected again in subsequent reinvigoration steps. As a result, when simulations are limited, the belief may converge prematurely to a restricted subspace, neglecting other valid regions of the state space.

To address these limitations, POMP-BE-PD introduces a modified belief update and resampling strategy. Initially, the belief is constructed by sampling particles from a uniform distribution over all valid states, namely, all possible object locations. We also introduce an auxiliary variable pp, which stores the set of object locations that have not yet been ruled out:

$$pp = \{j \in \mathcal{H} \mid j \text{ not yet observed}\},$$

initialized as $pp = \mathcal{H}$. At each time step, the agent receives observations via the object detector, and pp is updated by removing positions within the current field of view (FOV) that do not contain the target object. The belief is then re-sampled uniformly from the set of states consistent with pp, *i.e.*, $\mathcal{H} \setminus pp$. This method explicitly samples from the full set of object positions that have not yet been ruled out, independently

of previous belief states. This approach avoids the local feedback loop inherent in rejection sampling, ensuring a more comprehensive and stable exploration of the state space. As shown in the experimental results, this strategy consistently improves performance across a variety of scenarios.

Probabilistic Detection. Our agent is equipped with the Target Driven Instance Detector (TDID [4]), a model specifically designed to detect and classify individual instances of object categories. Given an input image, TDID outputs a list of bounding boxes (if any), each associated with a confidence score $s \in [0,1]$ and a class label c. In our setup, we consider only detections with confidence scores above 0.9. To evaluate the quality of detections, we rely on the standard metrics of True Positives (TP), False Positives (FP), and False Negatives (FN), and define the following performance metrics:

$$Recall = \frac{TP}{TP+FN} \qquad Precision = \frac{TP}{TP+FP}$$

Additionally, we compute the F_1 score, which is the harmonic mean of Precision and Recall:

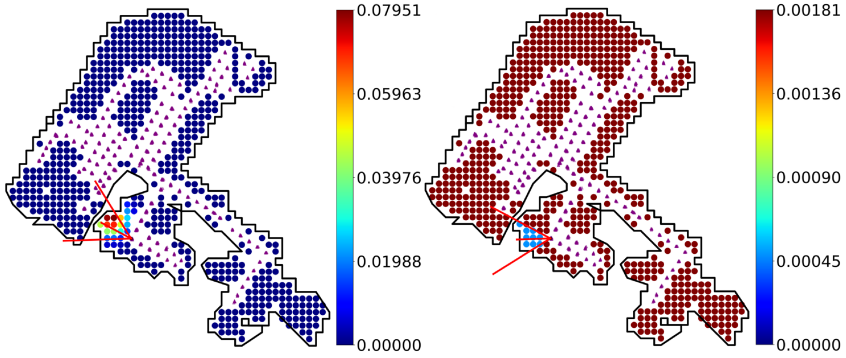
$$F_1\text{-score} = 2\frac{Precision \times Recall}{Precision + Recall},$$

where $F_1 \in [0,1]$. In POMP [132], the planner terminates the exploration phase as soon as the object detector identifies the target object within the agent's field of view (FOV). Thus, even a single false positive can prematurely end the exploration, preventing the agent from reaching the actual target object. In POMP-BE-PD, we aim to mitigate this limitation by not only relying on the current observation but also incorporating the full history of observations into the decision-making process.

We begin by defining a vector $\mathcal{D} = \{d_1, \ldots, d_k\}$, where each d_j represents the probability that the target object is located at position j, based solely on the object detector's output at the *current time step*. Thus, after each time step in the real world, we reset all values in $\mathcal{D}$.

We consider two distinct scenarios:

(i) If the object detector finds the target object within the current FOV, we set $d_j = 0$ for all locations j outside the FOV, and assign probabilities within the FOV according to a multivariate normal distribution with mean at the detected object location (see Fig. 3.3(a)).



(a) Object inside FOV (b) Object outside FOV

Fig. 3.3 The two cases considered when creating the vector D. Example derived from Home_003_2. In case (a), the objective is to determine the location of the object and assign probabilities in the form of a multivariate normal distribution. In (b), we assign low probabilities to the locations inside the FOV, and high probabilities to the locations outside it. Note: we assign different scales to the colorbar for ease of visualization.

(ii) If the target is not detected, we assign $d_j = F_1$ for locations inside the FOV, and $d_j = 1 - F_1$ for those outside the FOV (see Fig. 3.3(b)).

In both cases, the vector $\mathcal{D}$ is normalized so that $\sum_{j=1}^{k} d_j = 1$. Note that F_1 is class-specific, reflecting the detector's performance for the particular object class.

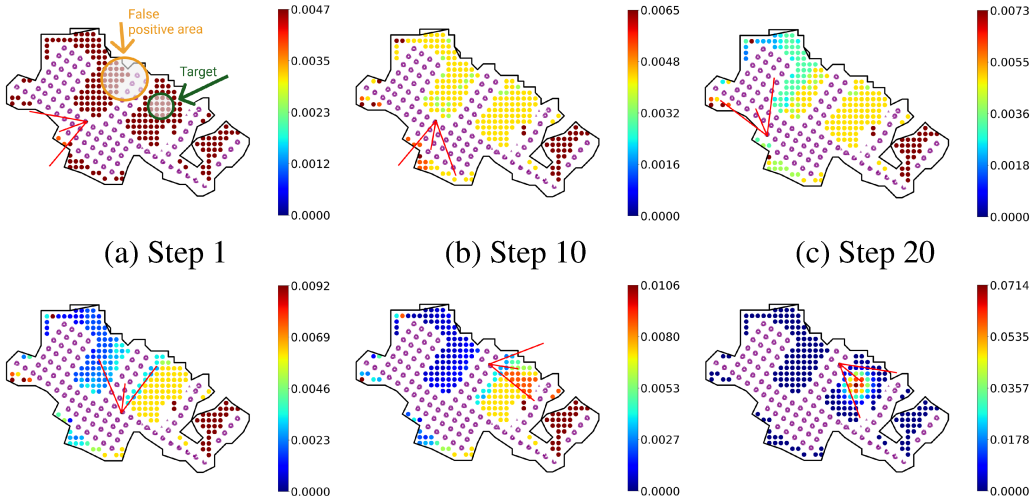
We also introduce an auxiliary vector $\mathcal{R} = \{p_1, \ldots, p_k\}$, which captures the probability of the target object being in each location j based on *the entire observation history*. At time $t = 0$, we initialize this distribution uniformly as $p_j^0 = 1/n$, where n is the total number of candidate locations. For each subsequent time step $t \geq 1$, the probabilities are updated using the rule:

$$p_j^t = \frac{p_j^{t-1} \cdot d_j^t}{\sum_{i=1}^{k} p_i^{t-1} \cdot d_i^t} \tag{3.1}$$

for all $j \in \mathcal{H}$.

We define a termination threshold $\tau = \frac{c}{n}$, where $c \in \mathbb{N}$ is a confidence constant, *i.e.*, it allows us to increase the confidence of our probabilistic detection. The exploration phase terminates when the probability p_j of a location j within the current FOV exceeds this threshold. Formally, the termination condition is:

$$(p_j \geq \tau) \wedge (L_{i,j} = 1). \tag{3.2}$$



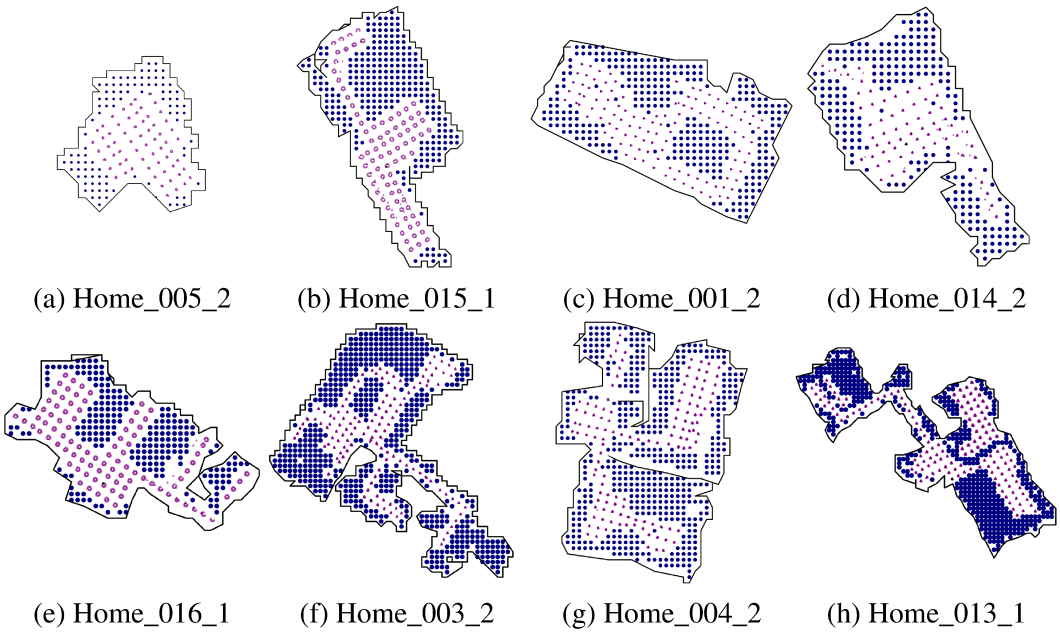
(a) Step 1 (b) Step 10 (c) Step 20

(d) Step 33 (e) Step 50 (f) Step 51

Fig. 3.4 Evolution of the probabilities p_j inside Home_016_1 using the proposed approach POMP-BE-PD. In step (a), we initialize the agent in the environment; we highlight the target position and a false positive area. From step (b) to (c), the robot explores the top area; in step (d) we show the robustness of our approach to a false positive; finally, in step (e), we identify highly probable locations, locating the target in step (f).

The update rule in Eq. 3.1 is a form of Bayesian inference, where the probability distribution is refined over time as new observations are incorporated. In our formulation, the distribution is not parameterized by a known function; rather, it directly represents the probability values over the candidate object locations. Bayesian inference is known to be optimal in minimizing the expected risk of incorrect decisions. Through this procedure, when the object is not observed in the current FOV, we increase the probabilities of it being elsewhere. On the other hand, if it is detected, we increase the likelihood in nearby positions and reduce it elsewhere. Importantly, this mechanism does not rely solely on the detector's output; instead, it aggregates information across time steps to build a more reliable estimate. An example episode illustrating this evolving probability distribution is shown in Fig. 3.4.

Probabilistic Docking. Given the object location $j \in \mathcal{H}$ that satisfies the exit condition in Eq. 3.2, we first determine the *destination pose*, *i.e.*, the agent's pose $\hat{i} \in \mathcal{G}$ that is both closest to the target location and oriented toward it. We then compute the shortest path from the current pose $i \in \mathcal{G}$ to the destination pose $\hat{i} \in \mathcal{G}$ using Dijkstra's algorithm [26]. While the agent moves along this path, the object detector



(a) Home_005_2 (b) Home_015_1 (c) Home_001_2 (d) Home_014_2

(e) Home_016_1 (f) Home_003_2 (g) Home_004_2 (h) Home_013_1

Fig. 3.5 Corresponding 2D floor maps (not in scale) for the test scenes from AVBD of 3 different difficulty levels (as in [106]). For each environment, we report the name. As the difficulty increases, we can note an increment of possible object locations and more difficult spatial layouts.

is disabled, as the confidence in the object's location is deemed sufficiently high. This approach outperforms the *Robust Visual Docking* strategy proposed in [132], which continues to rely on the object detector during navigation. In contrast, our method avoids distractions caused by false positives or missed detections, which could otherwise lead the agent away from the actual goal, particularly during the critical final approach.

3.2.2 Experimental Results

Dataset. We evaluate our agent POMP-BE-PD on the Active Vision Dataset Benchmark [3], a public benchmark for active visual search comprising over 30,000 RGB-D images collected across 15 indoor environments and featuring 33 target objects. Following the classification proposed in [106], we categorize each scene as *simple*, *medium*, or *hard* based on the complexity of the visual search task. A *simple* environment consists of a single small room, while a *medium* environment typically includes a large room or an additional small space such as a bathroom or an open

area. A *hard* environment consists of multiple large interconnected rooms. In our experiments, we select two simple apartments (Home_005_2 and Home_015_1), three medium apartments (Home_001_2, Home_016_1, Home_014_2), and three hard apartments (Home_003_2, Home_004_2, Home_013_1). Illustrative examples of these environments are shown in Fig. 3.5.

Metrics. We evaluate our approach using three metrics. The *Success Rate* (SR) [7], the primary metric in this work, is defined as the percentage of trials in which the agent successfully reaches one of the designated destination poses (as specified in AVDB), out of the total number of episodes. A higher SR indicates more effective search performance. The *Average Path Length* (APL) measures the average number of poses visited by the agent across successful episodes, divided by the total number of successful episodes. Lower APL values correspond to higher absolute efficiency. Finally, the *Success weighted by Path Length* (SPL) [7] is defined as:

$$SPL = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{l_i}{\max(p_i, l_i)}, \tag{3.3}$$

where N is the number of test episodes, l_i is the length of the shortest path between the start and goal for episode i, p_i is the actual path length taken by the agent, and S_i is a binary success indicator for episode i. Higher SPL values indicate better efficiency, balancing both success and path optimality. In this work, the term “efficiency" refers to the agent’s ability to explore and reach the target using the shortest possible path. An episode is considered successful if the agent reaches the destination pose defined by AVDB within a fixed number of steps (200 in our experiments) starting from the initial pose configuration defined in [3].

Baselines. We compare POMP-BE-PD against five baselines: *(i)* random walk, in which the agent, at each time step, randomly selects an action from the set of feasible actions; *(ii)* EAT [106]; *(iii)* DQN [71]; *(iv)* DQN-TAM [72]; and *(v)* POMP [132]. Among these, only POMP is an unsupervised online solver, whereas the others rely on training data to learn a policy. Due to the lack of publicly available code for the referenced methods, except for EAT, we follow the evaluation protocol introduced in [106] to report results. Compared to the original benchmark protocol defined in [3], this protocol restricts evaluation to ground truth (GT) annotations for object

detection and includes a reduced number of scenes and target objects. Notably, DQN and DQN-TAM are evaluated on only two scenes (one simple and one medium difficulty), making their average performance values unsuitable for a fair overall comparison. Furthermore, both methods adopt different subsets of target objects during evaluation, which further limits the comparability.

3.2.2.1 Results.

We compare POMP-BE-PD with state-of-the-art methods in Tab. 3.1, using ground truth object annotations and following the evaluation protocol defined in [106]. As shown, POMP-BE-PD achieves a higher SR than EAT, with a slight increase in Average Path Length (APL). We consider this increase reasonable, as the scenarios evaluated are more challenging, a point we will further discuss in the following sections. For completeness, Tab. 3.1 also includes a comparison with DQN. It is important to highlight that DQN requires training across 13 scenarios, whereas POMP-BE-PD operates without any training. Notably, DQN performs better in the easy scenario, while POMP-BE-PD achieves higher SR in the medium one, with comparable APL. Results obtained using the object detector from [4] are presented in Tab. 3.3 for both POMP and POMP-BE-PD. Notably, POMP-BE-PD achieves a significant improvement of 35% in both SR and SPL, primarily due to its enhanced capability to manage more challenging scenarios.

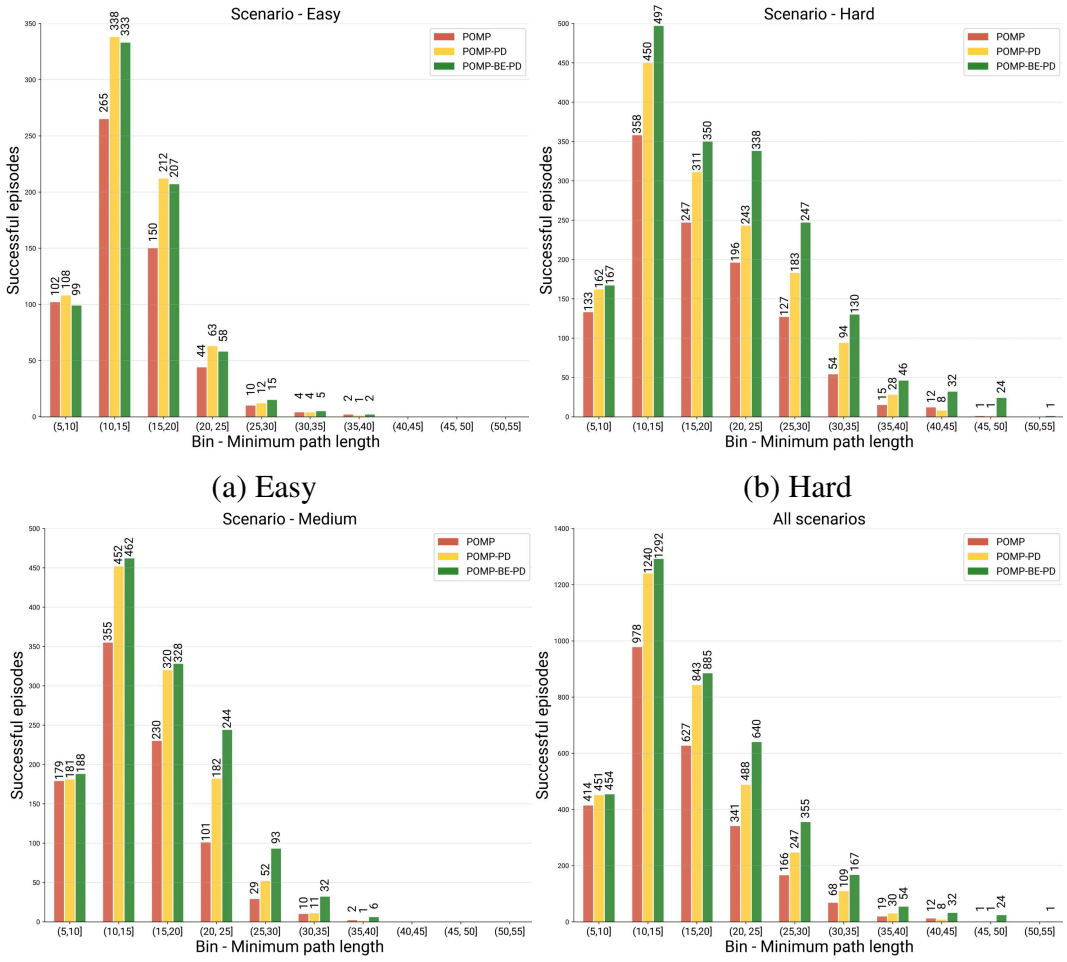
Table 3.1 Results on three scenes from AVDB using GT objects annotations. All methods are compared using the protocol defined in [106]. The asterisk $^{(*)}$ indicates that the evaluation is performed on a different subset of objects.

Method	Easy (Home_005_2)			Medium (Home_001_2)			Hard (Home_003_2)			Avg.		
	SR ↑	APL ↓	SPL ↑	SR ↑	APL ↓	SPL ↑	SR ↑	APL ↓	SPL ↑	SR ↑	APL ↓	SPL ↑
Random Walk	0.32	74.00	0.06	0.11	74.48	0.02	0.10	79.27	0.02	0.18	75.91	0.03
EAT [106]	0.77	12.20	0.42	0.73	16.20	0.56	0.58	22.10	0.41	0.69	16.80	0.46
DQN$^{(*)}$ [71]	*1.00*	*11.06*	-	*0.69*	*18.15*	-	-	-	-	-	-	-
DQN-TAM$^{(*)}$ [72]	*0.98*	*17.85*	-	*0.60*	*24.19*	-	-	-	-	-	-	-
POMP [132]	0.98	13.60	0.71	0.73	17.10	0.58	0.56	20.50	0.40	0.76	17.07	0.56
POMP-BE-PD	**0.98**	11.93	0.71	**0.80**	17.86	0.60	**0.92**	24.52	0.58	**0.90**	18.10	0.63

Ablation: Belief update. In the following, we analyze POMP-BE-PD through an ablation study, isolating key components. Specifically, we evaluate POMP-BE,

which corresponds to POMP equipped only with the new belief update mechanism. Specifically, we want to answer the following questions: *"Does the new belief update reduce the episode length? What are the benefits of the new belief update when navigating difficult scenarios?"*

In Fig. 3.6, we group the test episodes based on their difficulty level and categorize them by the minimum path length required to reach the target. Specifically, Fig. 3.6(a) shows the results for easy scenes (Home_005_2 and Home_015_1); Fig. 3.6(b) presents data for hard environments (Home_003_2, Home_004_2, Home_013_1); Fig. 3.6(c) includes medium-difficulty cases (Home_001_2, Home_014_2, and Home_016_1); and finally, Fig. 3.6(d) aggregates all the scenarios available in AVDB. These visualizations suggest that excluding already observed locations from the belief update enhances the efficiency of the exploration phase, leading to more effective search behavior.



(a) Easy (b) Hard

(c) Medium (d) All

Fig. 3.6 We aggregated the episodes by the minimum number of steps to reach the object, thus incorporating the difficulty of the episode. In Figure (a) the results for the Easy scenarios; in Fig. (b) Hard Scenarios; in Fig. (c) the Medium ones and finally, in Fig. (d), the sum of all scenarios. Results using the object detector provided by [4], both during planning and docking. Focusing on the POMP-PD method (yellow bar), we can observe the increment of efficiency and efficacy due to the introduction of the Belief Update (green bar), since both methods do not change the exit condition during planning (Probabilistic Detection).

Tab. 3.2 further evaluates the contribution of our belief update mechanism by isolating potential sources of error. In this case, ground truth object annotations are used in place of the detector, thereby eliminating both false positives and missed detections during planning and docking. In the easy setting, performance differences are minimal: SR drops slightly (by 0.01), while APL decreases and overall efficiency,

Table 3.2 Result of different versions of improved POMP with more scenes per difficulty level in AVD. POMP-BE is POMP with the improved Belief Update. Result using the ground truth annotations instead of the detector, using 2^{10} simulations during the planning phase. The new Belief Update consistently increase the efficiency of the exploration phase, thus reducing the Average Path length, and increasing the SR and SPL.

Difficulty	Scene	POMP[132]			POMP-BE		
		SR ↑	APL ↓	SPL ↑	SR ↑	APL ↓	SPL ↑
Easy	Home_005_2	0.94	12.96	0.73	0.93	12.26	0.72
	Home_015_1	0.75	23.66	0.45	0.73	17.04	0.52
	Avg.	**0.84**	18.31	0.59	0.83	14.65	0.62
Medium	Home_001_2	0.80	18.20	0.57	0.81	19.95	0.55
	Home_014_2	0.76	41.07	0.38	0.90	19.99	0.55
	Home_016_1	0.71	29.64	0.39	0.83	36.55	0.50
	Avg.	0.76	29.64	0.45	**0.85**	25.50	0.53
Hard	Home_003_2	0.43	21.90	0.27	0.79	31.93	0.45
	Home_004_2	0.45	66.20	0.17	0.57	47.71	0.28
	Home_013_1	0.55	49.72	0.27	0.74	53.11	0.41
	Avg.	0.48	45.94	0.24	**0.70**	44.25	0.38
Average		0.67	32.92	0.40	**0.79**	29.82	0.50

measured by SPL, improves. We speculate that this is due to the simplicity of the layouts (see Fig. 3.5(a), 3.5(b)) and the narrow distribution of minimum path lengths (see Fig. 3.6(a)), which limit the advantage offered by the improved belief update. Conversely, as the scenario complexity increases, with a larger number of possible object positions and more intricate spatial layouts, the benefits of the new strategy become more evident. In the medium difficulty setting, SR improves from 0.76 to 0.85, APL drops from 29.64 to 25.50, and SPL increases from 0.45 to 0.53. A comparable improvement is observed in the hard scenarios as well.

Ablation: Probabilistic Detection & Docking. In this section, we want to answer the following question: "*Does Probabilistic Detection reduce the number of false positives? Is there a way to improve the docking, also considering the knowledge gathered during the planning?*"

To address the first research question, we categorize and examine the types of failures encountered across episodes. Specifically, we define three categories of errors:

(i) *Localization error*: this occurs when the POMCP exploration phase terminates due to the exit condition being satisfied, but the target object is not within the agent's field of view.

(ii) *Docking error*: this happens when the agent correctly detects the object at the end of the exploration phase but fails to reach one of the designated *successful destination poses* defined by AVDB.

(iii) *Other error*: this includes all remaining failure modes, such as the agent exceeding the step limit without detecting the object, or performing actions not allowed along the planned path.

Fig. 3.7 Percentage of error of POMP, POMP-PD and POMP-BE-PD, averaged over all scenarios. The errors are categorized into three types: Localization, Docking and Other. We used the object detector provided by [4], during both planning and docking.

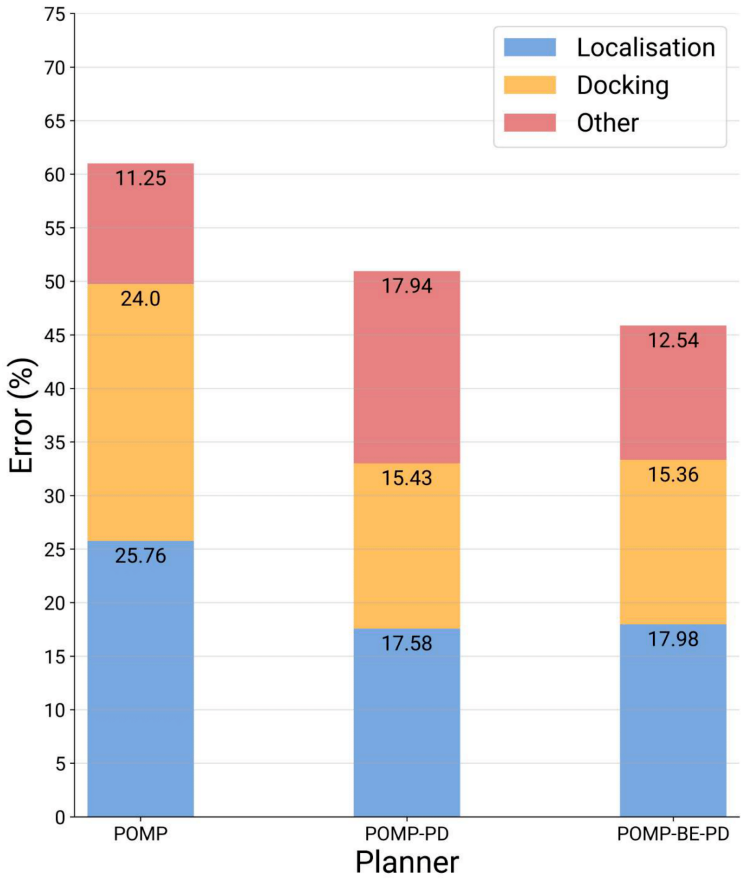


Fig. 3.7 presents the average distribution of these error types for each planner across all test scenarios. A key observation is that incorporating the Probabilistic Detection

(POMP-PD) mechanism significantly reduces false positives—by approximately 32%, improving the method's robustness. The revised Belief Update, on the other hand, yields a notable reduction (around 30%) in errors grouped under the "Other" category, which reflects an increase in the agent's decision-making efficiency. Additionally, leveraging planning-phase information during the docking phase contributes to greater robustness. This is evidenced by the substantial drop in Docking errors (roughly 35.7%).

To further assess the impact of the Probabilistic Detection component, we conduct an ablation study in Tab. 3.3. This isolates the effect of probabilistic modeling from the belief update mechanism, applying the detector during both planning and docking stages. Across all difficulty levels, the variant POMP-PD achieves a significant increase in Success Rate (SR), ranging from 19% to 25% over the original POMP method, while maintaining comparable Success weighted by Path Length (SPL). Although we note a rise in Average Path Length (APL), this is expected: enhancing confidence in object detection, and in particular, reducing the impact of false positives, requires additional exploratory steps. To this end, the agent must accumulate sufficient evidence to elevate the probability of the target location above the threshold τ, resulting in longer trajectories.

Table 3.3 Results of POMP and variations of POMP-BE-PD with more scenes per difficulty level in AVDB [3] using the object detector provided by [4].

Difficulty	Scene	POMP[132]			POMP-BE			POMP-PD			**POMP-BE-PD**		
		SR ↑	APL ↓	SPL ↑	SR ↑	APL ↓	SPL ↑	SR ↑	APL ↓	SPL ↑	SR ↑	APL ↓	SPL ↑
Easy	Home_005_2	0.60	17.90	0.40	0.58	16.18	0.41	0.81	26.08	0.42	0.79	22.70	0.45
	Home_015_1	0.49	34.76	0.22	0.45	38.76	0.23	0.55	35.34	0.23	0.54	30.50	0.26
	Avg.	0.54	26.33	0.31	0.52	27.47	0.32	**0.68**	30.71	0.33	0.67	26.60	0.35
Medium	Home_001_2	0.40	20.73	0.24	0.39	19.36	0.24	0.50	31.00	0.24	0.57	28.50	0.31
	Home_014_2	0.53	47.60	0.25	0.60	18.52	0.38	0.60	45.79	0.24	0.66	21.38	0.37
	Home_016_1	0.29	50.23	0.12	0.28	47.05	0.13	0.36	57.73	0.12	0.41	53.26	0.16
	Avg.	0.41	39.52	0.20	0.42	28.31	0.25	0.49	44.84	0.20	**0.55**	34.38	0.28
Hard	Home_003_2	0.19	26.60	0.10	0.33	30.53	0.18	0.39	62.36	0.13	0.48	42.86	0.20
	Home_004_2	0.42	69.84	0.15	0.55	47.31	0.26	0.44	70.26	0.14	0.54	61.93	0.20
	Home_013_1	0.25	61.41	0.12	0.31	77.09	0.14	0.26	62.80	0.09	0.34	54.38	0.15
	Avg.	0.29	52.62	0.12	0.40	51.64	0.19	0.36	65.14	0.12	**0.45**	53.06	0.18
Average		0.40	41.13	0.20	0.44	36.85	0.25	0.49	48.92	0.20	**0.54**	39.44	0.27

Case Study: Evolution of Probabilities. Fig. 3.4 illustrates a complete episode executed by our proposed method, highlighting both the dynamic evolution of the

location probability distribution and the system's resilience to false positives. The initial pose of the agent is shown in Fig. 3.4(a). From there, the agent begins exploring the upper part of the environment, as visualized in Fig. 3.4(b) and 3.4(c), but no detections occur during this phase. A critical situation arises in Fig. 3.4(d), where the agent experiences a false positive detection. Thanks to the Bayesian update rule defined in Eq. 3.1 and the resulting probabilistic map, the agent correctly avoids prematurely ending the search. In contrast, using the original POMP formulation from [132], the episode would have been incorrectly terminated at this point due to the deterministic observability model.

Additionally, we observe an increase in the probabilities associated with unexplored regions (the right-hand section of the map). This reflects the system's inference that, in the absence of detections, the target is more likely to reside in unvisited locations. Finally, in Fig. 3.4(e), the agent identifies a region with a high likelihood of containing the object. This culminates in the successful discovery of the target, as shown in Fig. 3.4(f).

3.2.3 Conclusion

We introduced POMP-BE-PD, a novel approach for Active Visual Search (AVS) in environments with known 2D floor maps. Built upon the POMCP planning framework, POMP-BE-PD performs online policy learning by leveraging topological map information, thereby eliminating the need for data-intensive training procedures. To address the challenges posed by unreliable object detectors, particularly false positives and missed detections, we replaced the standard deterministic detection with a *probabilistic* formulation. After each action in the environment, Bayesian inference over a probability distribution of candidate object locations, resulting in a 32% reduction in false positives. Furthermore, to overcome the limitations of traditional POMCP belief updates in AVS, we proposed a new belief update strategy that maintains a uniform distribution over all unexplored candidate locations. This change improves exploration robustness and particle reinvigoration. Extensive evaluation on the AVDB benchmark demonstrates that POMP-BE-PD achieves state-of-the-art performance. Through detailed ablation studies, we showed that each component contributes to performance gains: on average, we observe a 35% increase in success rate and a 4% reduction in path length compared to the earlier method, POMP.

Chapter 4

Agent Robustness via Instruction Understanding

Interacting with robots using natural language instructions is one of the most challenging long-term goals of embodied AI. To foster progress in this area, a range of tasks have been proposed, each testing a different aspect of language-guided perception and navigation capabilities. For instance, the Vision-and-Language Navigation (`VLN`) task requires agents to follow detailed, multi-step natural language instructions such as: "*Exit the bedroom. Turn left and go directly into the bathroom. Stop there*." Other tasks like ObjectNav (`ObjectNav`) and Instance ObjectNav (`InstanceObjectNav`) instead focus on goal-directed autonomous search. In `ObjectNav`, the goal is to find an object of a given category (*e.g.*, "*Find a chair*"), while `InstanceObjectNav` involves locating a specific object *instance* described in natural language (*e.g.*, "*Find the wooden piano located near the curtain in the bedroom*").

Despite growing progress on these benchmarks, a significant limitation persists: many natural instructions provided by humans are often imperfect, either due to errors, such as incorrect references or contradictions, or because they are underspecified, relying on implicit assumptions or contextual knowledge. This chapter addresses these challenges through two complementary contributions: *Error-Aware Instruction Following* (Section 4.1) and *Ambiguity-Aware Navigation* (Section 4.2).

4.1 IEDL/I2EDL

The first line of investigation focuses on navigation under erroneous instructions, as shown in Fig. 4.1.

Fig. 4.1 An agent navigates in a scene, following instructions expressed in natural language, for example *"Exit the bathroom and go left (✓right), then turn left at the big clock and go into the bedroom and wait next to the bed."* By just changing "right" to "left" in the instruction, the agent terminates the exploration in the wrong location, ignoring the fact that along the path it did not see the "big clock" (yellow arrow).

We begin by formally categorizing the types of natural language instruction errors commonly encountered in the VLN setting, including mistakes involving *directional* cues, *room* references, *object* mentions, or combinations of these elements. During our preliminary study, we show that agents that are not equipped to handle such imperfections often fail to complete the task, as in Fig. 4.2. To systematically study this problem, we introduce the *Detection and Localization of Instruction Errors* task, which requires agents to identify if an error exists in the instruction, and to localize which part of the instruction may have caused the error. Building on this, we propose the *Instruction Error Detection & Localizer* (*IEDL*) method, a cross-modal transformer that jointly encodes vision and textual observations. It is equipped with dedicated heads for both error detection and localization, enabling instruction error detection and fine-grained localization. This is further extended in the *Interactive VLN in Continuous Environments* (IVLN-CE), simulating the real-world cases where

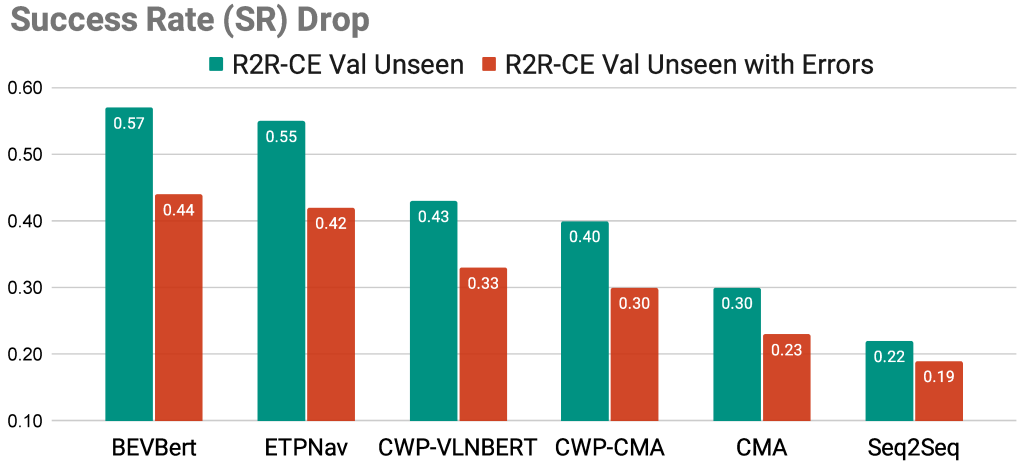


Fig. 4.2 Comparison of the Success Rate (SR) of different methods (in order [5, 6, 43, 43, 57, 57]) working on continuous environments. We show the SR on the standard R2R-CE dataset split `Val Unseen` (green) and the drop in SR performance when errors are present (red). Interestingly, we see up to -25% drop in SR when up to three errors among *{Direction, Room, Object}* per episode are present.

humans are allowed to make mistakes when providing instruction, where agents are allowed to ask clarification questions. This models real-world use cases where users may make mistakes and agents must interactively resolve them. We then propose an effective baseline, *Interactive Instruction Error Detection & Localizer* (I2EDL), which interacts with the user in an online manner upon detecting instruction errors, prompting them with specific questions to lower user cognitive load. Finally, we propose a unified evaluation metric that balances success rate (SR) with user interaction number, providing a single scalar value to compare agent performance while accounting for both task effectiveness and interaction efficiency.

4.1.1 Related Works

In the following, we review existing approaches to Vision-and-Language Navigation (VLN), focusing on both standard and failure-aware methods. Additionally, we discuss related research on trajectory-instruction (vision-language) alignment, which is crucial for effective navigation in language-guided tasks.

Vision-and-Language Navigation. Introduced by Anderson et al. [8], the VLN task, also referred to as Room-to-Room (R2R), has become a cornerstone benchmark in embodied AI. The original setup leveraged the Matterport3D simulator and dataset [18], which provides real 360-degree RGB-D scans structured as a discrete, undirected navigation graph. Early approaches in this setting relied on attention mechanisms [19, 42, 20] or recurrent neural networks [8, 131, 30] to encode the agent's visual history and ground it in natural language instructions. Instead, in [57], VLN in Continuous Environment (VLN-CE) is introduced, in which agents are allowed to move freely, thus removing the assumption of known environment topologies, short-range oracle navigation and perfect agent localization. In doing that, they translated the nav-graph R2R trajectories to the continuous environments in the Habitat simulator [105], providing a more challenging and real-world grounded scenario. To address the gap between discrete and continuous VLN settings, Hong et al. [43] proposed a prediction mechanism that generates candidate waypoints to guide the agent during navigation. Due to the long time horizon of episodes in this setting, more recent approaches have adopted spatial representations such as metric maps [133] and topological memory modules [6, 5] to represent observations' history. The current state-of-the-art, BEVBert [5], introduces a hybrid mapping strategy that supports both long-term planning and short-term local reasoning, alongside a map-based pretraining paradigm. Differently, we shift attention toward a new direction: introducing and tackling the *Detection and Localization of Instruction Errors* problem within the VLN framework.

Failure Analysis on VLN. A number of prior studies have focused on analyzing the behavior and failure modes of VLN agents. For instance, Zhu et al. [151] highlights that agents utilize both directional and object-related tokens when making navigational decisions, and that transformer-based agents acquire a better cross-modal understanding of objects and display strong numerical reasoning abilities. Similarly, Hahn et al. [39] assesses the influence of spatial and directional language cues by selectively masking specific token types, such as directions, nouns, or numerical references, in path-ranking models to study their impact on decision-making. Another study [146] identifies a significant drop in agent performance in unknown environments, attributing this to biases in low-level visual features. Complementing these insights, Yang et al. [139] propose a framework to assess agent behavior based on distinct skills by applying targeted interventions and observing variations in ac-

tion outputs, with a particular focus on stop actions, directional cues, and object- or room-related instructions. Distinct from these prior works, our study focuses on the impact of erroneous instructions on VLN performance. Instead of simply masking tokens [39], we actively modify instructions by injecting specific types of errors. Moreover, unlike works that assess model generalization across environments [146] or explore skill-based intervention analysis [139], our investigation targets the agent's robustness to natural language inconsistencies within the instruction itself.

Instruction-Trajectory Alignment. Aligning instructions with corresponding navigation trajectories is a critical capability in Vision-and-Language Navigation (VLN), ensuring that semantic cues in both modalities are correctly interpreted by the agent. Huang et al. [45] introduces the Cross-Modal Alignment (CMA) task, which involves distinguishing between valid instruction-path pairs and corrupted ones. Notably, the original instructions remain intact and only the paths are artificially modified. Zhao et al. [147] take a complementary approach by assessing the quality of instructions produced by VLN instruction-generation models such as [30, 121]. Through an evaluation involving human wayfinders, they propose a compatibility model that classifies instructions as either high or low quality based on their effectiveness in guiding human agents to the correct goal. Notably, they go beyond earlier work by introducing perturbations to instructions, including direction swap, swapping entities within the same phrase, removing, duplicating and shuffling sub-sentences in the same instruction. A contrastive learning framework is proposed in [67] to enhance the generalization of navigation policies. Their approach augments training data by modifying instructions through synonym substitution, contextual insertion, and back-translation. Positive and negative examples are carefully constructed based on their semantic proximity to the original instruction. Negative samples are created by rearranging instruction components to generate incoherent or misleading directions. While all these works aim to improve instruction-path alignment to strengthen VLN *policy learning*, our focus diverges in two significant ways. First, rather than enhancing alignment, we study how instruction inaccuracies affect navigation performance, introducing a new task. Second, our experiments are grounded in the more realistic continuous environment framework, as opposed to the majority of prior work, which is constrained to discrete graph-based environments.

Interactive VLN. In [21], a dedicated policy allows the agent to query the oracle when uncertain, penalized by a negative reward. The oracle responds with the next optimal action. Mistakes are simulated by injecting oracle errors, whereas we begin with instructions that already contain mistakes. In [89], help is requested when the agent is lost or uncertain. The oracle then provides short-term goals via natural language, either through direct intervention or indirect hints. A policy is trained to manage the budgeted queries. [94] addresses Audio-Visual-Language Embodied Navigation (AVLEN), where the agent operates under a limited number of oracle queries. While interaction frequency is not explicitly analyzed, performance is measured through standard success metrics. Nguyen and Daumé III [88] further relaxes oracle assumptions, simulating assistants that can only help when the agent is within their attention zone. Interaction is measured by the number of help requests per task, and contains multimodal data.

4.1.2 Task & Benchmark

We introduce the Vision-and-Language Navigation in Continuous Environments (`VLN-CE`) task. Next, we define the types of instruction errors considered in our work, and finally, we present our proposed task, Detection and Localization of Instruction Errors.

Vision-and-Language in Continuous Environment. We define the natural language instruction provided to the agent as $\mathcal{I}$, consisting of F words, *i.e.*, $\mathcal{I} = \{w_1, \ldots, w_F\}$. At each time step t, the agent receives an RGB-D observation O_t, representing the visual input captured during navigation. We denote the sequence of such observations across a navigation episode as $\mathcal{O} = \{O_1, \ldots, O_T\}$, where T is the total number of steps executed by a navigation policy π. The objective in the `VLN-CE` task is to learn a policy π that, at each step, interprets the instruction $\mathcal{I}$ along with the current observation O_t to select an appropriate low-level action from the set $a_t \in \{$`Forward 0.25m, Turn Left 15°, Turn Right 15°, STOP` $\}$.

Instruction Error Types. Human-provided navigation instructions are often prone to mistakes, either due to misremembering spatial layouts [44], general confusion, or imperfect memory, as shown in Fig. 4.1. To systematically study the effects of such

inaccuracies on navigation performance in `VLN-CE`, we classify instruction errors based on their semantic roles and real-world plausibility, including:

(i) *Direction Errors.* Directional terms like *left*, *right*, *forward*, or backward are critical for guiding agents. However, these terms are also commonly confused with their antonym due to their binary and relative nature, *e.g.*, *right/left* or *into/out of.* We define a *direction error* as any instance in which at least one correct direction word in the instruction is replaced by an incorrect one.

(ii) *Object Errors.* Objects mentioned in navigation instructions provide important visual anchors. However, humans may misidentify or confuse objects, particularly those frequently co-located in common space. For example, confusing a sofa with a chair is more likely than confusing a toilet with a chair. An object error is defined as the incorrect substitution of an object class with another. In our study, we focus on plausible confusions influenced by common co-occurrence (*i.e.*, *common sense*) patterns in household scenes.

(iii) *Room Error*. Rooms serve as key contextual waypoints in navigation. Due to memory inaccuracies or spatial similarity, users may confuse adjacent rooms, such as saying bathroom instead of bedroom. A room error occurs when at least one room reference is incorrectly replaced with another. We consider errors that reflect room adjacency priors, such as *"Go into the bedroom (✓ bathroom). Stop in front of the cabinet, near the plant."*

(iv) *Room & Object Error*. Since both room and object errors stem from similar causes, we also consider cases where both types occur. A Room & Object Error is defined when the instruction includes at least one incorrect room reference and one incorrect object mention.

(v) *All Error*. Finally, we define an all error category, where direction, room, and object errors appear together in a single instruction.

Task Definition. We now define the *Instruction Error Detection and Localization* task. Given a natural language instruction $\mathcal{I}$, which describes the steps required to reach a target location, and a sequence of visual observations $\mathcal{O}$ collected by a VLN agent operating under a specific policy π, the goal of the *Instruction Error Detection*

task is to learn a function:

$$d_\pi : \mathcal{I} \times \mathcal{O} \rightarrow \{\text{True}, \text{False}\}$$

that returns True if the instruction contains any errors, and False otherwise.

If an error is detected, *i.e.*, $d_\pi(\mathcal{I}, \mathcal{O}) = \text{True}$, the next step is to perform *Instruction Error Localization*. This task focuses on identifying the specific positions within the instruction where the erroneous words appear. More formally, the localization function is defined as:

$$l_\pi : \mathcal{I} \times \mathcal{O} \rightarrow \mathcal{P}(\{0, 1, \ldots, len(\mathcal{I}) - 1\})$$

where $\text{len}(\mathcal{I})$ denotes the number of words in the instruction, and $\mathcal{P}(\cdot)$ refers to the power set operator, returning all possible subsets of word positions in $\mathcal{I}$.

Benchmark Definition: R2RIE-CE. We introduce the first benchmark for *R2R with Instruction Errors in Continuous Environments* (`R2RIE-CE`). Built on top of the R2R-CE benchmark [57], `R2RIE-CE` is designed to evaluate `VLN-CE` methods under the presence of erroneous instructions. R2R-CE consists of three splits: `Train`, `Val Seen` and `Val Unseen`. Specifically, we focus on the most challenging `Val Unseen` split, which includes 1839 evaluation episodes with novel paths, instructions and scenes not seen during training.

To generate instruction containing errors, we introduce artificial perturbations based on the error types defined in the Detection and Localization of Instruction Errors task. For each error type, we construct a dedicated split based on the `Val Unseen` split. We begin by filtering out episodes that lack words relevant to the targeted error category, *e.g.*, episodes without directional terms are excluded from the direction error split. Next, we remove episodes whose instructions have a path length shorter than a predefined threshold τ words, ensuring sufficient instruction complexity. The remaining set of valid, unmodified episodes forms our base set of correct episodes, denoted as $\mathcal{E}_\mathcal{C}$.

Then, for each episode $e_i \in \mathcal{E}_\mathcal{C}$, we generate a corresponding erroneous instance by perturbing the instruction according to the defined error type. The perturbations are applied as follows:

(i) *Direction Errors.* We consider a curated set of frequently occurring directional terms in instructions, including: *left/right*, *leftmost/rightmost*, *inside/outside*, *into/out of*, *forward/backward*, *go down/go up*, *go around/go back*. A direction error is introduced by replacing the original directional word with its antonym.

(ii) *Object Errors.* We first extract a vocabulary of common object categories $\mathcal{C}$ that frequently appear in natural language instructions, filtering out synonyms. For each object class $c_i \in \mathcal{C}$, we define a co-location set $\mathcal{C}_i$ consisting of object classes that are typically found in the same room. To inject an object error, we replace c_i with a random class $c_j \in \mathcal{C}_i$.

(iii) *Room Error*. We define a list of common room types $\mathcal{R}$ found in indoor settings, such as *bedroom, kitchen, archway, bathroom, living room, lounge, hallway, dining room, office, gym, laundry* and *restroom*. Each room $r_i \in \mathcal{R}$ is associated with a set of spatially adjacent rooms $\mathcal{R}_i$, derived using ConceptNet relationships and manually verified pairs. A room error is introduced by replacing r_i with a randomly selected room $r_j \in \mathcal{R}_i$.

(iv) *Room & Object Error*. We introduce both an *Object* and *Room* error.

(v) *All Error*. We introduce both a *Direction* and *Room & Object* error.

We then defined the set of perturbed episodes, for each type of error, as $\mathcal{E}_\mathcal{P}$. For every perturbed episode $e_i \in \mathcal{E}_\mathcal{P}$, we store both the error type and the exact position of the altered word as metadata. This information serves as ground truth for evaluating the performance on the Detection and Localization of Instruction Errors task. As a result, for each category of instruction error, we construct two corresponding sets: the clean set $\mathcal{E}_\mathcal{C}$ and the perturbed set $\mathcal{E}_\mathcal{P}$, maintaining a 50% ratio. Tab. 4.1 summarizes the statistics of the validation splits generated for each error type.

Table 4.1 Statistics of the R2RIE-CE benchmark.

Error type	Episodes #	Errors per episode	Mean instr. length (tokens)
Direction	3218	1	31.59
Room	2064	1	32.05
Object	3162	1	30.94
Room & Object	1734	2	33.35
All	1586	3	34.58

4.1.3 Method

Our method *Instruction Error Detection & Localizer* (IEDL) is built on top of BEVBert [5], a state-of-the-art `VLN-CE` policy. In the following, we will refer to policy π as an optimal BEVBert policy. Note again that π is frozen, thus it will not receive any parameter updates.

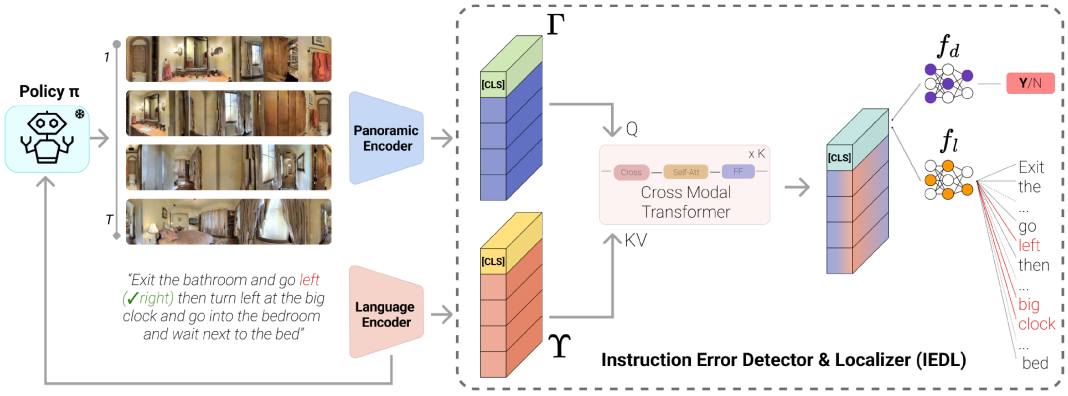


Fig. 4.3 Architecture of our proposed *IEDL* model, representing the scenario depicted in Fig 4.1. The frozen policy π follows Instruction Υ, producing a sequence of observations $\mathcal{O}$. Then, a panoramic encoder and a language encoder produce, respectively, the trajectory visual features Γ and instruction features Υ. We then feed the trajectory set Γ and Υ to a cross-modal multilayer transformer to produce visual-language aligned representations. Finally, two task-specific heads perform *Instruction Error Detection* and *Instruction Error Localization*.

Model Structure. We show the full IEDL architecture in Fig. 4.3. We represent instruction embeddings as $\Upsilon \in \mathbb{R}^{W \times D}$, where the instruction $\mathcal{I}$ is tokenized and padded to a fixed length of $W = 80$ tokens, following standard practice [5, 6]. Each token is embedded using a BERT-based text encoder [25], with D denoting the embedding dimensionality.

The visual trajectory is denoted as $\Gamma = \{V_1, ..., V_T\}$, where each $V_t \in \mathbb{R}^D$ represents the embedding of the panoramic visual observation O_t at time step t. These embeddings correspond to the sequence of nodes selected by the navigation policy π in response to the encoded instruction Υ.

Following the approach in [5], each embedding V_t is derived by extracting features from a pre-trained Vision Transformer (ViT-B/16-CLIP [96]) and processing them through a panoramic encoder [19] to obtain contextual visual representations. For each episode i, we collect the trajectory $\Gamma \in \mathbb{R}^{T \times D}$, which consists of the sequence of visual embeddings generated by policy π. To preserve positional information, we apply sinusoidal positional encoding. Additionally, similar to the `[CLS]` token in BERT [25], we prepend a learnable embedding `[CLS]` $\in \mathbb{R}^D$ to Γ. This token is later used for downstream predictions.

We integrate the instruction embeddings Υ with the trajectory embeddings Γ using a cross-modal transformer with k layers (see Fig. 4.3, right). Our architecture is inspired by [120] but omits the bi-directional cross-attention component. In each transformer layer, cross-attention is applied with the trajectory as the query (Q) and the instruction as key-value pairs (K, V), followed by self-attention and a feed-forward block. The `[CLS]` token, which is now enriched with fused visual-linguistic representations, is passed through two classification heads to perform the two downstream tasks:

(i) *detection head:* the trajectory-instruction alignment head $f_d : \mathbb{R}^D \to \mathbb{R}$ estimates the alignment score $\sigma(a) \in [0, 1]$ via the function d_π where $\sigma(\cdot)$ denotes the sigmoid function;

(ii) *localization head*: the error localization head $f_l : \mathbb{R}^D \to \mathbb{R}^W$, implementing l_π, identifies erroneous tokens within the instruction.

Both heads consist of a multilayer perceptron (MLP) followed by a ReLU activation, LayerNorm, and a final MLP.

Training procedure. The model is optimized using two distinct loss functions. The trajectory-instruction matching head f_d is trained with a Binary Cross-Entropy loss, denoted as $\mathcal{L}_d$. In contrast, the error localization head f_l is supervised using a standard Cross-Entropy loss $\mathcal{L}_l$. Since an instruction may contain multiple errors, we sum the losses for each localization term. The overall objective function combines the two losses as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_d + \frac{\lambda_2}{E} \sum_{i=1}^{E} \mathcal{L}_l$$

Here, E denotes the number of actual errors in the instruction, while λ_1 and λ_2 control the relative weighting of each loss component.

4.1.4 Experimental Results

Metrics. We adopt standard evaluation metrics for VLN, following established protocols [8]. An episode is considered successful if the agent finishes within 3 meters of the goal. The primary VLN metrics are Success Rate (SR) and Success weighted by Path Length (SPL).

To evaluate performance on *Instruction Error Detection*, we use the Area Under the ROC Curve, as in (AUC) [147], which measures the area under the True Positive Rate vs. False Positive Rate curve. In this work, an instruction is labeled as positive if it contains at least one error. Moreover, AUC serves as the primary evaluation metric (highlighted in grey).

For assessing *Instruction Error Localization*, we introduce a new metric called *Absolute Token Distance* (ATD). This metric quantifies the discrepancy between predicted and ground-truth positions of perturbed tokens. Specifically, for an episode i, let ℓ^i_j be the ground-truth index of the j^{th} erroneous token, and $\hat{\ell}^i_j$ its predicted index. The ATD is defined as follows:

$$ATD = \frac{1}{N_{\mathcal{E}_P}} \sum_{i=1}^{N_{\mathcal{E}_P}} \frac{1}{J_i} \sum_{j=1}^{J_i} |\ell^i_j - \hat{\ell}^i_j|, \tag{4.1}$$

where $N_{\mathcal{E}_P}$ is the number of perturbed episodes, and J_i is the number of perturbed tokens in episode i. Lower values indicate more accurate localization of instruction errors.

Implementation details. We refer readers to [5] for details on BEVBert. In our setup, we use $k = 4$ layers for the cross-modal transformer and set the feature dimension to $D = 768$. The loss weights λ_1 and λ_2 are both empirically set to 1. We train the IEDL model for up to $9,000$ iterations using the AdamW optimizer.

All training experiments are conducted using the *All Error* benchmark type, *without* common sense priors, from the R2R-CE training set. This setting excludes enforced object co-location and room adjacency from the room set, ensuring no additional commonsense bias is introduced during training. Model selection is based on the best `AUC` score achieved on the *All Error* benchmark *with* commonsense, evaluated on the `Val Unseen` split. The selected model is then used for evaluation on all other benchmarks.

Baselines. Since no existing baselines are available for this task, we compare IEDL against two alternatives: a random baseline and a zero-shot alignment baseline:

(i) *Random.* For each episode i, the instruction embedding Υ_i is randomly classified as either correct or incorrect. If labeled as incorrect, we randomly predict J_i token positions, where each $\hat{\ell}^i_j$ is sampled uniformly from $[0, \text{len}(\Upsilon_i)]$. The value J_i corresponds to the number of errors expected in the dataset (see Tab. 4.1).

(ii) *CLIP Alignment.* This zero-shot baseline uses CLIP [96] for visual-semantic alignment. For each instruction $\mathcal{I}$, we extract a set of room and object tokens $\mathcal{K}$ using an off-the-shelf POS tagger [13]. At each time step t, we query CLIP to obtain the top-k predicted room and object labels from observation O_t, aggregating them into a set $\mathcal{S}$ of visually grounded entities. CLIP prompts follow the format: "*a photo of a: <room or object>*", based on the room and object vocabularies described. The instruction is flagged as erroneous if $\mathcal{K} \nsubseteq \mathcal{S}$, meaning instruction tokens $\mathcal{K}$ are not visually grounded during navigation. Error localization is then performed by retrieving the indices of instruction tokens in $\mathcal{K}$ that are not found in $\mathcal{S}$ as : $\{\hat{l} \,|\, k_{\hat{l}} \in \mathcal{K} : k_{\hat{l}} \notin \mathcal{S}\}$.

Effect of errors on VLN agent's performance. We begin by investigating how different error types impact the performance of a `VLN` policy in continuous environments, measured by `SR` and `SPL`. To quantify the effect of each error type, we compute the drop in `SR` between perturbed and unperturbed conditions. Specifically,

Table 4.2 Results of our proposed *IEDL* method on our proposed benchmark. We show the SR and SPL metrics of the frozen policy, and the drop in SR performance when errors are present (ΔSR %). We then analyze the classification (AUC) and Localization (ATD) performance of different methods. Error types with ⋆ indicate benchmark with common sense. AUC is highlighted as it is the main metric.

Origin split	Error type	Policy [5]			Random		CLIP Align-ment		**IEDL**	
		SR ↑	**SPL** ↑	$\Delta_{SR}(\%)$	**AUC** ↑	**ATD** ↓	**AUC** ↑	**ATD** ↓	**AUC** ↑	**ATD** ↓
R2R-CE Val Unseen	Direction	0.53	0.43	-18.64	0.50	10.54	0.50	11.05	0.58	8.13
	Room⋆	0.58	0.49	-6.66	0.50	11.03	0.57	9.63	0.80	7.73
	Object⋆	0.56	0.46	-8.47	0.51	10.94	0.59	8.76	0.74	9.21
	Room&Object ⋆	0.57	0.47	-11.47	0.49	11.56	0.64	7.98	0.91	7.34
	All ⋆	0.52	0.43	-30.64	0.51	12.22	0.63	8.68	0.94	6.14
	Avg.	0.55	0.46	-15.17	0.50	11.26	0.59	9.22	**0.79**	**7.71**

for each error type, we evaluate the policy on the correct episodes $\mathcal{E}_C$ and their corresponding perturbed versions $\mathcal{E}_P$. The relative decline in success rate is calculated as:

$$\Delta_{SR}(\%) = SR(\mathcal{E}_C) - SR(\mathcal{E}_P)$$

Since $\mathcal{E}_P$ is derived directly from $\mathcal{E}_C$, this metric isolates the impact of each error type on navigation performance. Results are summarized in Tab. 4.2 under the "Policy" column (SR, SPL, and $\Delta_{SR}(\%)$ columns). Among all error types, the *Direction* error set leads to the most significant performance degradation, with a relative drop of -18.64% in SR. The *Object* error with common sense causes a decrease of -8.47%, while the *Room* error with common sense results in a -6.66% decrease. These results align with prior findings [151], indicating that VLN agents strongly rely on directional and object tokens. However, in contrast to [151], we observe that in continuous environments, agents are more sensitive to directional errors than to object-related ones (-18.64% vs. -8.47%). We attribute this to the environment property of the discrete R2R environment. Specifically, its "navigation graph", which may introduce strong implicit biases [57]. Interestingly, combining *Room* and *Object* errors with common sense yields a performance drop of -11.47%, which is smaller than the sum of their individual effects, suggesting overlapping information redundancy. The most severe decline is observed under the *All* error type with common sense, which combines all perturbations and results in a -30.64% drop in success rate. Overall, these results, especially for the *Direction* error set, highlight a critical vulnerability in current VLN-CE models and emphasize the need for greater robustness to linguistic

perturbations, particularly those affecting directional grounding.

Does the instruction contain an error? In this experiment, we evaluate the classification head f_d against two baselines using the AUC metric. Results are reported in Tab. 4.2.

The random baseline serves as a reference point to detect dataset bias and define a performance lower bound. Since the dataset is balanced with 50% correct ($\mathcal{E}_{\mathcal{C}}$) and 50% perturbed ($\mathcal{E}_{\mathcal{P}}$) episodes, its expected AUC is approximately 0.50.

We also include CLIP Alignment as a zero-shot baseline that simulates a human-like approach, *i.e.*, verifying whether instruction content is visually grounded in the agent's observations. Notably, CLIP Alignment does not require any training. CLIP Alignment is relatively effective on *Object* and *Room* error types, achieving a AUC of 0.64 when both are present (*Room&Object*). However, it performs no better than random in the case of *Direction* errors. This likely occurs because even when directional errors appear late in the instruction, CLIP Alignment may still successfully ground earlier object and room references (*i.e.*, $\mathcal{K} \subseteq \mathcal{S}$), masking the effect of the error.

Our proposed *IEDL* achieves the highest AUC across all benchmarks, demonstrating robust alignment detection. However, a lower AUC of 0.58 on the *Direction* error type reveals that this category remains particularly challenging. On average, *IEDL* significantly outperforms both baselines, achieving a mean AUC of 0.79, compared to 0.50 for Random and 0.59 for CLIP Alignment.

Can we localize the error? This experiment assesses the effectiveness of the localization head f_l in *IEDL* for identifying erroneous tokens within an instruction. Results in terms of ATD are also presented in Tab. 4.2.

The random baseline yields a mean ATD of 11.26. In contrast, the CLIP Alignment baseline consistently outperforms random across all error types, proving particularly effective in the *Room&Object* benchmark.

IEDL achieves the best localization performance overall, outperforming both baselines across all benchmarks, with the exception of the *Object* category, where CLIP Alignment performs slightly better. However, although CLIP Alignment localizes object-related errors well, its corresponding AUC is lower (0.59) compared

to *IEDL* (0.74), indicating that it likely detects only a subset of the erroneous instructions.

Notably, the average ATD of *IEDL* is 7.71, which is roughly the length of a typical sub-sentence in the R2RIE-CE instructions. This suggests that *IEDL* is capable of localizing errors with sub-sentence-level granularity.

Can *IEDL* detect errors in the R2R-CE dataset? This experiment explores the use of a pre-trained *IEDL* as a semi-automated tool for identifying potentially mislabeled episodes, including those in the original R2R-CE dataset. We apply the alignment classification head f_d of *IEDL* to the Val Unseen split of R2R-CE and isolate episodes with alignment scores a exceeding a threshold of $\tau_a = 0.99$.

Interestingly, *IEDL* flags 25 out of 1839 episodes as highly likely to contain inconsistencies. Manual inspection reveals that 8 of these 25 cases indeed correspond to incorrect ground-truth annotations. This result indicates that current evaluations in VLN-CE may be partially affected by annotation errors, highlighting the need for more robust dataset validation.

Can *IEDL* generalize to other datasets and models? To assess generalization, we apply the trained *IEDL* to a different navigation policy ETPNav [6] and a different dataset, RxR-CE [60]. Compared to R2R-CE, RxR-CE features significantly longer and more detailed instructions (averaging 110 tokens vs. 30). Without any fine-tuning, we run *IEDL* on episodes with English (en-IN and en-US) instructions, using the same threshold $\tau_a = 0.99$ as before.

Out of 3669 episodes, the model flags 118 as potential outliers. Upon manual review, 10 of these episodes were confirmed to contain errors, including directional inconsistencies, object mismatches, incorrect goal annotations, or mesh issues, suggesting they should be excluded from the validation set. These results demonstrate that *IEDL* can generalize across models and datasets and could serve as a valuable tool for large-scale dataset verification.

4.1.5 I2EDL: Interactive IEDL

IEDL detects and localizes errors in an offline mode, *i.e.*, the errors in the instructions are identified only after an agent has finished its trajectory, thus leaving no chance for the agent and the user to interact and recover the errors *while exploring the environment*. However, enabling human-agent interaction during navigation could be extremely effective, since agents may prompt the user for instruction corrections, thus improving the success rate of the task. Such interactive `VLN` with error awareness introduces two main additional challenges:

(i) *interaction timing:* the agent should identify potential errors promptly at an early stage, with only partial observations of the scene. If the detection is delayed in time, it may be too challenging for the agent to correct its trajectory.

(ii) *interaction number:* since it is not ideal to have an agent that constantly interacts with a human user, asking for potential errors (both for human disturbance and cognitive load), it is essential to have an accurate online instruction error detector and localizer. Essentially, *"asking the right question at the right time."*

In this section, we introduce the *Interactive VLN in Continuous Environments* (*IVLN-CE*), in which human users are allowed to make errors in their initial instructions and subsequently correct them if the agent accurately detects and locates the errors through human-agent interactions. We then propose an effective baseline, the *Interactive Instruction Error Detection & Localizer* (I2EDL) method, which is based on IEDL and operates in an online mode with partial observation. Finally, we propose the *Success weighted by Interaction Number* (`SIN`) metric, which reflects both the navigation performance and interaction effectiveness, by encouraging a higher success rate while limiting the interaction numbers.

Task Formulation. We extend the *Detection and Localization of Instruction Errors* task, presented in Section 4.1.2, to the interactive setting.

For every episode i, and at every step t, the agent has the possibility to query the human, checking if a particular token ℓ_i^j is correct or not, where $j \in [0, len(\Upsilon_i) - 1]$ and $len(\cdot)$ returns the total number of tokens for instruction Υ_i. Asking just a token to the user would be ineffective, since the user would hardly understand the sense of a single word (token), and, in the case of multiple instances of the same word,

misunderstandings could easily arise. Therefore, the agent passes to the user a portion (*context*) of the instruction, made by multiple tokens $L_{j,\varsigma_l} = [\ell_i^{j-\varsigma_l}, \ell_i^{j+\varsigma_l}]$, where ς_l is the *contextualization length*. The human, upon receiving a request from the agent, returns the correct token if a wrong token is found within $[\ell_i^{j-\tau_l}, \ell_i^{j+\tau_l}]$, where τ_l is a *localization threshold*. This correction mechanism ensures that the human can provide the correct token even if there is a slight discrepancy of τ_l tokens in the location pointed out by the agent.

I2EDL. For each episode i, we execute the policy π following instruction $\mathcal{I}_i$ for at least p steps to acquire the set of visual observations $\mathcal{O} = \{O_1, ..., O_p\}$. When the current step $t \geq p$, we use the detection head f_d of IEDL to check if the alignment score is $a \geq \tau_d$, where $\tau_d = 0.6$ is a *detection threshold*. If the detection is positive, meaning that the instruction contains at least one error, we use the IEDL localization head f_l to localize the errors. Formally, we apply the $softmax$ operator over the output of f_l, and then select the token ℓ_i^j with the highest probability, where j is the index of the token. When a positive detection occurs, we increment variable NI_i, showing that the agent has detected and localized errors in the instruction. We then simulate an agent-human interaction by asking the human the following question:

> *"I think there is an error in this part of the instruction: <part>, and specifically on this <token>. Is this the case?"*

In this question, *<token>* refers to token ℓ_i^j, while *<part>* refers to context L_{j,ς_l}. If the range identified by tokens $[\ell_i^{j-\tau_l}, \ell_i^{j+\tau_l}]$ contains the error, then the agent receives the correct token, and the embedding for instruction Υ_i is recomputed. Otherwise, if the detection is a false positive or no error is found by the human within the specified range, no action is performed by the human. After the interaction, the agent resumes its navigation, having the possibility to query the user for every step until T steps are performed or the action `STOP` action is selected by the policy, meaning that the agent believes it has reached the goal.

Success weighted by Interaction Number. Since our interaction scheme for VLN is new, we need to propose a novel figure of merit. The rationale is that we want to weight the success rate, depending on how often the agent requires the human

intervention: the higher the number of interventions, the less valuable the success rate.

We thus propose SIN, *i.e. Success weighted by Interaction Number*, specifically designed to combine, in a single measure, both the SR and the number of interactions with the user. Inspired by the Success weighted by Path Length metric [7], we define SIN as:

$$SIN = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{1}{1+\lambda \frac{NI_i}{max(1,NE_i)}} \tag{4.2}$$

where NI_i is the number of interactions with the user, NE_i is the number of errors in episode i and S_i is a binary indicator of success for episode i. The $max(\cdot)$ operator in the denominator ensures the number of interactions NI_i is weighted by the number of errors NE_i. Note that if no errors are present for episode i, $max(1,NE_i) = 1$. λ is a weighting factor that modulates the penalty for the number of interactions We consider SIN as the primary metric in evaluating methods addressing the *IVLN-CE* task.

SIN properties. SIN ranges between 0 and 1. A higher value indicates better navigation performance and interaction efficiency. Moreover, the proposed SIN metric possesses several favorable properties:

(i) when no interaction is performed with the human (*i.e.*, when NI_i is 0), SIN is mathematically equivalent to SR.

(ii) SIN penalizes false positives detections.
Proof: for every correct episode i (*i.e.*, the instruction is correct and thus $NE_i = 0$), a perfect agent will not interact with the human, thus $NI_i = 0$. If this is not the case, NI_i is increased accordingly, thus minimizing the SIN metric. Note that, in this scenario, the denominator is 1, resulting in increased importance assigned to each unnecessary interaction.

(iii) SIN penalizes repetitive interactions.
Proof: for every incorrect episode i (*i.e.*, the instructions contain error), the SIN metric will be penalized as the agent requests multiple interactions with the human.

(iv) the weighting factor λ prevents the denominator from becoming excessively large. The metric can still show the improvement in SR while penalizing

excessive interaction. We found that a $\lambda = 0.01$ is a good compromise between weighting SR and the number of interactions.

4.1.5.1 Experimental Results

Metrics. We evaluate interactive agents with SR, SPL, the newly introduce SIN metric and the *Mean Interaction Number* (MIN), defined as:

$$MIN = \frac{1}{N}\sum_{i=1}^{N} NI_i$$

where NI_i is the number of times the agent interacts with the user in episode i.

Baselines. To the best of our knowledge, there are no published baselines since IVLN-CE is a novel task. To this end, we compare I2EDL with two baselines:

(i) *"Random Interaction"*: for every episode i and for every step, it randomly predicts if instruction Υ_i contains errors. If the detection is positive (*i.e.*, the instruction contains an error), we then randomly predict a token ℓ_i^j, where $j = rand(0, len(\Upsilon_i) - 1)$ and *rand* returns a random number between the arguments.

(ii) *"Always Ask"*: this baseline prompts an interaction at every step from step $p = 4$ onward. At each interaction, it randomly predicts the erroneous token in the same way as the *"Random Interaction"* baseline.

Table 4.3 Results show the increase of SIN (in %) under different paradigms of interaction on R2RIE-CE benchmark, with localization threshold $\tau_l = 1$, weighting factor $\lambda = 0.01$ from step $p = 4$ onwards. The primary metric SIN is highlighted. Under the *"No Interaction"* column, we report the SR, SPL metrics of the BEVBert policy [5], also showing the Success Rate Upper Bound ($\overline{\text{SR}}$). For *I2EDL*, we set detection threshold $\tau_d = 0.6$. Error type based on R2RIE-CE *Val Unseen* Dataset.

Error type	No interaction			Random Interaction				Always Ask				**I2EDL**			
	SR ↑	SPL ↑	$\overline{\text{SR}}$ ↑	SIN ↑	MIN ↓	SR ↑	SPL ↑	SIN ↑	MIN ↓	SR ↑	SPL ↑	SIN ↑	MIN ↓	SR ↑	SPL ↑
Direction	53.4	43.5	58.5	52.9	1.82	53.6	43.6	52.8	3.64	54.3	44.0	**53.2**	0.50	53.4	43.5
Room*	58.1	48.6	60.4	57.1	1.81	57.9	48.4	56.8	3.62	58.4	48.9	**58.1**	0.79	58.4	48.8
Object*	56.1	46.1	58.7	55.8	1.75	56.6	46.4	55.2	3.53	56.7	46.7	**56.1**	0.70	56.3	46.3
Room&Object *	57.3	46.9	61.1	56.9	1.86	57.4	47.3	57.3	3.75	58.4	47.8	**58.3**	1.15	58.5	47.7
All *	52.4	42.6	61.9	53.3	1.97	53.8	43.5	53.2	3.95	54.1	43.8	**53.8**	1.37	54.0	43.4
Avg.	55.4	45.5	60.1	55.2	1.84	55.9	45.8	55.0	3.70	56.4	46.2	**55.9**	0.90	56.1	46.0

Success Rate and interaction paradigm. In Tab. 4.3, under the *"No Interaction"* column, we first report the SR and SPL of the current state-of-the-art method [5], establishing a lower bound without interaction.

In the IVLN-CE setting, the goal is to minimize user interactions while maximizing their impact. Therefore, in Tab. 4.3 we report the results of the two baselines. The *"Random Interaction"* baseline triggers interactions at random and cannot distinguish correct from erroneous instructions, resulting in ~ 2 interactions per episode. This leads to user fatigue and low SIN, which penalizes unnecessary interactions.

The *"Always Ask"* baseline further highlights this trade-off: it achieves high SR by constantly interacting, but with a high MIN (3.70) and low SIN.

Our method, *I2EDL*, outperforms both. First of all, we note that I2EDL has a much higher SIN than the *"Random Interaction"* , meaning that our method is able to detect instruction errors and localize them more precisely, thus maximizing the effectiveness of the interactions. This is also reflected under the SR column, in which, apart from the *Direction* error benchmark, *I2EDL* has an equal or better SR performance, while halving the average MIN (0.90 *vs* 1.84) and scoring consistently lower in terms of SIN. Compared to the *"Always Ask"* baseline, *I2EDL* has a higher SIN (55.9 *vs* 55.0) while having an extremely low MIN score of 0.90 *vs* 3.70. Notably, *I2EDL* has the lowest results on the *Direction* error benchmark, indicating the challenge of R2RIE-CE.

What is the SR upper bound on R2RIE-CE? To estimate the upper bound SR ($\overline{\text{SR}}$) on R2RIE-CE, we simulate a perfect agent unaffected by instruction errors. For each perturbed episode $i \in \mathcal{E}_p$, we replace the faulty instruction with the correct one. Episodes in $\mathcal{E}_c$ remain unchanged. Tab. 4.3 shows the results. The largest gain appears in the *All* benchmark (52.4 $\rightarrow$ 61.9), where episodes contain all three error types. The second largest gain is in *Direction* (53.4 $\rightarrow$ 58.5). On average, $\overline{\text{SR}}$ improves by 8.48% (55.4 $\rightarrow$ 60.1).

Can agents recover from instruction errors? We evaluate the agent's ability to recover from instruction errors occurring at different time steps. To simulate this, for each perturbed episode $i \in \mathcal{E}_p$, the agent navigates using the perturbed instruction for t steps, then receives the correct instruction at step $t+1$ via a simulated human correction:

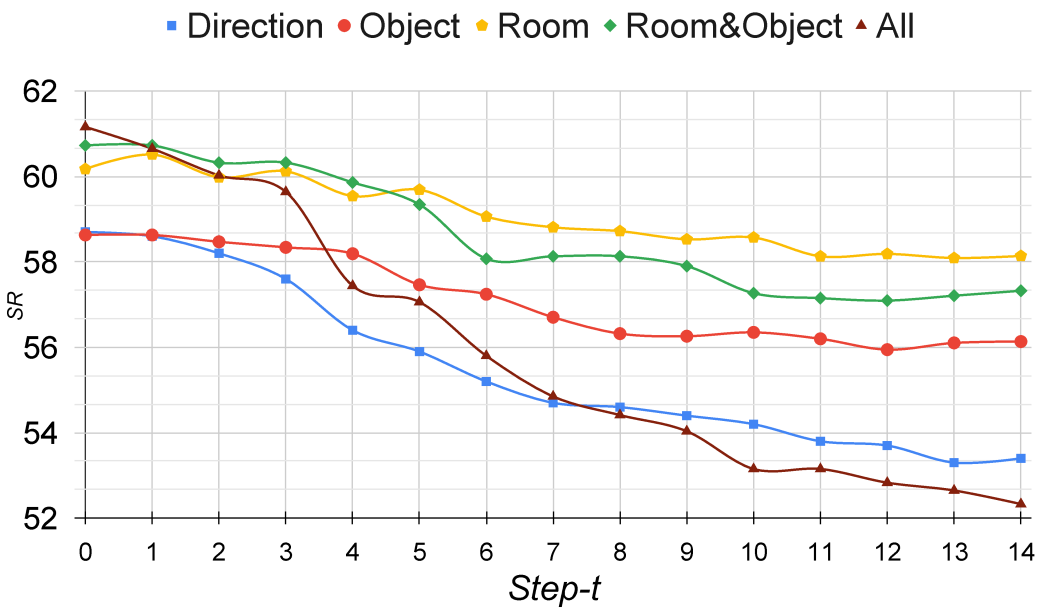


Fig. 4.4 Success Rate upper bound ($\overline{\texttt{SR}}$) at different step-t.

> *"Sorry, the instruction I gave you before is wrong. This is the correct one: <instruction>"*

where *<instruction>* is the correct instruction. Episodes in $\mathcal{E}_c$ remain unchanged. Fig. 4.4 reports SR as a function of step t for each benchmark in R2RIE-CE. The *Direction* and *All* benchmarks show the steepest SR drop as t increases, reflecting their high sensitivity to early errors, especially since they contain one and up to three instruction errors, respectively. Early correction is thus critical for recovery.

How do SIN and SR evolve over steps? This experiment analyzes how SR and SIN evolve over time under different interaction paradigms. In Fig. 4.5, we report SR (solid lines) and SIN (dashed lines) at different step-t values for: *(i) "Random Interaction"*, *(ii) "Always Ask"*, and *(iii) I2EDL* with $\tau_d = 0.6$ and $\tau_l = 1$.

As expected, *"Always Ask"* achieves the highest SR by constantly prompting the user. However, this results in low SIN due to excessive, unnecessary interactions. A similar pattern is observed for *"Random Interaction"*. In contrast, *I2EDL* achieves a better trade-off between task success and interaction efficiency with a higher SIN. These behaviors are correctly reflected by our proposed SIN metric.

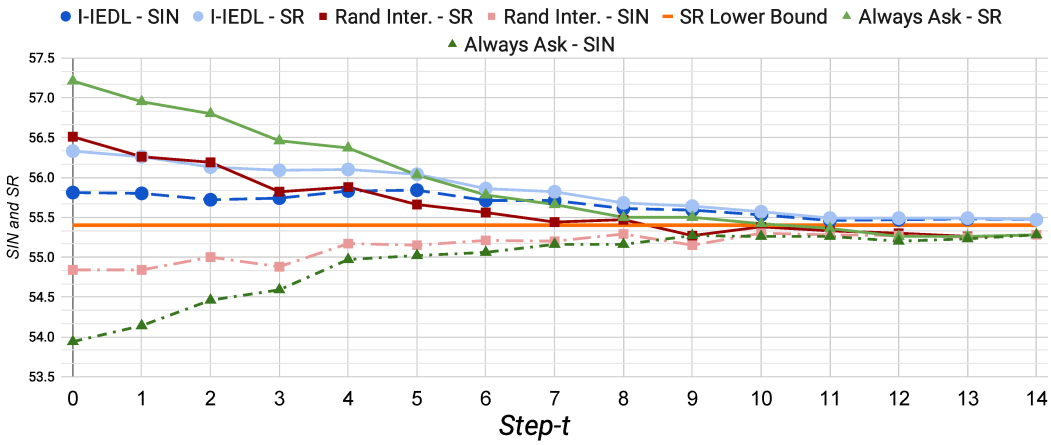


Fig. 4.5 SR and SIN plotted at different step-t for localization threshold $\tau_l = 1$. Specifically, dashed lines indicate the value for the SIN metric, while solid lines indicate the SR. The *"Always Ask"* baseline always interacts with the user from step-t onwards.

4.1.6 Conclusion

We introduced R2RIE-CE, a new benchmark for VLN-CE in which various types of errors are systematically injected into navigation instructions. Our experiments demonstrate that current state-of-the-art VLN-CE models are significantly affected by such instruction perturbations. Building on R2RIE-CE, we proposed the novel task of *Detection and Localization of Instruction Errors*, along with our method *Instruction Error Detection & Localizer* (IEDL), which consists of detection and localization heads. (IEDL) is capable of identifying and localizing errors at the sub-sentence level within the original instruction. We further demonstrated the practical relevance of our method by applying it to R2R-CE and RxR-CE, identifying 8 and 10 wrongly annotated episodes, respectively, that should be removed from the evaluation set.

To enhance robustness at test time, we introduced *IVLN-CE*, where agents can interact with users to correct instructions during navigation. We proposed a strong baseline, I2EDL, which performs online error detection and localization. Compared to baselines, I2EDL improves navigation success under erroneous instructions while minimizing unnecessary user interactions. Together, our benchmark, task formulations, and methods move towards more robust and adaptive VLN-CE agents capable of handling real-world linguistic errors.

4.2 AIUTA

Language-driven instance object navigation (`InstanceObjectNav`) assumes that a human initiates the task by providing a detailed description of the target to the embodied agent. The instance description typically contains nuanced details about the intrinsic (*e.g.*, color, material) and extrinsic (*e.g.*, context, spatial relations) attributes of the searched object instance, which are essential to *uniquely identifying* the target amid visual ambiguity. Although this description is crucial, providing it prior to navigation can be demanding for humans, as users may be unable or unwilling to supply all details in advance.

The second line of research thus explores navigation in the presence of ambiguous object descriptions. First, we introduce the *Collaborative Instance object Navigation* (CoIN) task, a novel problem setting in which the agent must actively resolve uncertainties regarding the target instance through *template-free, open-ended dialogues* with humans during navigation. A sketched illustration of this task is shown in Fig. 4.6. To tackle this challenge, we propose *Agent-user Interaction with UncerTainty Awareness* (AIUTA), a training-free framework that operates independently of the navigation policy and focuses on the human-agent interaction reasoning using Vision-Language Models (VLMs) and Large Language Models (LLMs). AIUTA equips the agent with two key components: a *Self-Questioner* and an *Interaction Trigger*. The Self-Questioner leverages an LLM and VLM in a self-dialogue process: it first describes the agent's observation, then extracts additional relevant details, while a novel entropy-based technique reduces hallucinations and inaccuracies, producing a refined detection description. The Interaction Trigger then uses this refined description to determine whether to ask the user a clarifying question, proceed with the current navigation plan, or stop the exploration altogether. To support systematic evaluation, we introduce CoIN-Bench, a curated benchmark focused on complex multi-instance environments, specifically designed to test agents under object ambiguity conditions. Notably, CoIN-Bench enables evaluation with both real and simulated human interactions. To simulate human responses, we leverage a Vision-Language Model (VLM) that has access to a high-resolution image of the target object. Through extensive experimentation, we show that this setup allows for reproducible and scalable evaluation of human-agent interactions.

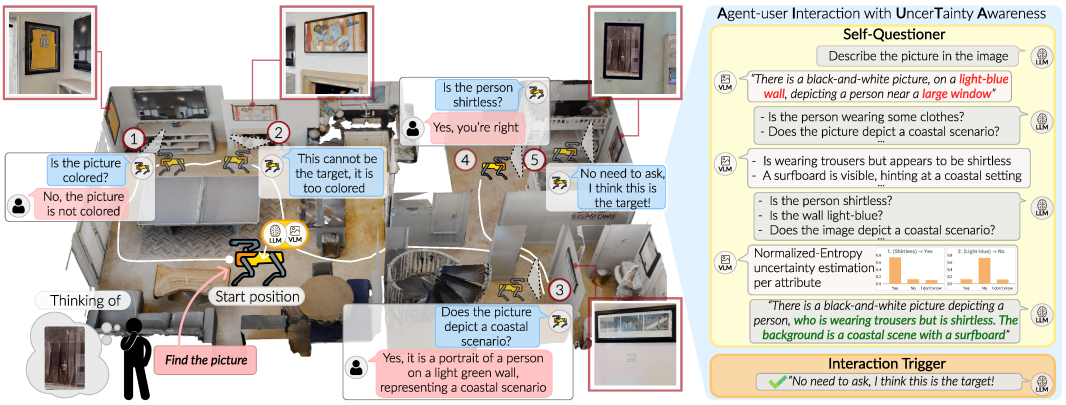


Fig. 4.6 Collaborative Instance object Navigation (CoIN) task illustration. A human provides a request (*"Find the picture"*) in *natural language*. The agent has to locate the object in a *completely unknown* environment *without any target image as input*, interacting with the user only when needed via *template-free*, *open-ended natural-language dialogues*. Our method, **A**gent-user **I**nteraction with **U**ncer**T**ainty **A**wareness (**AIUTA**), minimizes user interactions by equipping the agent with two modules: a **Self-Questioner** and an **Interaction Trigger**. The **Self-Questioner** leverages an LLM and VLM in a self-dialogue to describe the agent's observation and then extract additional relevant details, with a novel entropy-based technique to reduce **hallucinations and inaccuracies**, producing a refined **detection description**. The **Interaction Trigger** uses this refined description to decide whether to pose a question to the user (①,③,④), continue the navigation (②), or halt the exploration (⑤).

4.2.1 Related Works

We identify three main topics related to this section, which are briefly surveyed below.

Instance Object Navigation. `InstanceObjectNav` extends the traditional Object-Goal Navigation (`ObjectNav`) task [7], where the goal is to find a *specific* object instance rather than any instance of a given category. While prior work has used target images to specify the instance [58], we instead focus on user-provided natural language descriptions. Recent policies addressing this task fall into two broad categories: training-based [55, 100, 78, 112, 28, 142] and zero-shot [32, 149, 144, 61, 141, 143, 116]. Training-based methods typically use reinforcement learning [55, 112, 78], sometimes combined with behavioral cloning [100]. These methods often rely on vision-language-aligned embeddings to process goal modalities. For example, GOAT-Bench [55] uses CLIP-based goal embeddings, while others [78, 112] are trained on image-goal navigation tasks but evaluated on `ObjectNav`. Zero-shot methods, on the other hand, often extend the seminal

frontier-based exploration [138] by integrating large language model (LLM) reasoning [149, 143, 144, 61], CLIP-based localization [32], or vision-language semantic maps [115, 141] for frontier selection. Most recently, [10] proposed using multi-modal target specifications (a set of images and textual descriptions). In contrast, our approach enables natural human-agent interaction during navigation *without* requiring access to any target image.

Interactive Embodied AI. Human-agent interaction using natural language is a longstanding goal in Embodied AI. Common approaches allow agents to request assistance from users [117], with responses typically taking the form of shortest-path action sequences [111, 21] or concise natural language sub-goals that guide navigation [88, 89, 76, 94, 101]. In [103, 87], the authors propose a framework that estimates the uncertainty of an LLM-based planner, enabling the agent to either decide on the next action or query the user for help. Huang et al. [47] demonstrates that LLMs can generate inner monologues when provided with environmental feedback, which improves planning in robotic control tasks. Both [33, 123] present dialog-guided task completion benchmarks based on human-annotated question-answer pairs collected via Amazon Mechanical Turk. In [90], agent queries are restricted to three fixed request types. Similarly, [34] uses templated questions focusing on appearance, location, and direction. Zhu et al. [152] employs fixed-format queries such as "Should I go [dir] to the [obj]?", while [70] adopts a multiple-choice Q&A format based on rigid templates. ELBA [109] generates both oracle-based and model-based templated questions using ground-truth answers. FindThis [79] restricts the agent to selecting candidate object images as responses, without the ability to ask questions or use free-form natural language. In [23], the ZIPON task requires agents to find personalized objects. However, goals are manually annotated, and the user, simulated via an LLM, can only respond with predefined ground-truth information. Both [79, 23] assume access to pre-built top-down semantic or occupancy maps to locate objects of interest. In [87], help requests are allowed only at specific locations, after which the simulator provides a natural language subgoal and an image of the target. Zhang and Choi [145] relies on manually defined disambiguation intents for triggering clarifications, which limits applicability in open-world settings. In contrast, our agent identifies the target instance solely through *open-ended, template-free natural language dialogue* with the user without predefined response structures,

image inputs, or handcrafted disambiguation rules.

Vision-Language Models Uncertainty. Hallucinations, biases, reasoning failures, and the generation of unfaithful outputs are well-documented issues in large language models (LLMs) [49]. Orgad et al. [91] shows that truthful information tends to cluster around specific tokens, a property that can be exploited to improve error detection. However, such detectors often fail to generalize across datasets. Similarly, recent works reveal systematic limitations in vision-language models (VLMs) [127, 73], which may hallucinate or provide incorrect answers when faced with misleading or unanswerable questions [95]. To address this, PAI [73] enhances visual grounding by amplifying attention weights over image tokens. Zhao et al. [148] shows that a linear probe over the distribution of initial token logits is used to classify visual questions as answerable or unanswerable. Instead, CLARA [93] estimates LLM uncertainty through context sampling, distinguishing between certain and uncertain instructions. Zhu et al. [150] introduces VLM-LLM dialogues for image captioning. In contrast, we employ self-dialogue in an embodied task setting, where both the agent's observation and target information are used to generate clarification questions. Moreover, we introduce a novel uncertainty estimation mechanism, which reduces hallucinations and improves grounding.

4.2.2 Task & Benchmark

Task Definition. Collaborative Instance object Navigation (CoIN) defines a new variant of the `InstanceObjectNav` task in which an agent must navigate an unfamiliar 3D environment to locate a specific target instance. Unlike prior works, the agent collaborates with a human user through *open-ended*, *template-free*, and *natural language* interactions. Crucially, the agent autonomously decides when to engage the user to acquire necessary information about the target during navigation. The primary goal of CoIN is to identify the correct object instance while minimizing human input, thereby reducing the user's effort of providing detailed descriptions.

At the beginning of each episode, the agent is randomly placed in an unseen 3D scene [97]. Navigation begins when the user issues a natural language request I, which may be as brief as an open-set category label c, for example: "*Find the* $<$*category*$>$". Importantly, the agent does not receive any visual reference of the

target instance. We assume the user is: *(i)* fully informed about the target's details, and *(ii)* cooperative, providing truthful responses when queried by the agent.

At each time step t, the agent receives a visual observation O_t (*i.e.*, an RGB-D frame) and selects an action a_t from the set $A = \{$`Forward 0.25m, Turn Left 15°, Turn Right 15°, STOP, ASK` $\}$, where `ASK` is the novel action of CoIN. This action allows the agent to pose a free-form question $q_{a \to u}$, in natural language, to request additional details about the target from the user. Upon receiving the user's response $r_{u \to a}$, the agent updates its internal set of *facts*, *i.e.*, a collection of attributes and descriptors of the target instance denoted as F_t. This update is formalized as $F_t = F_{t-1} \cup r_{u \to a}$.

Navigation concludes when termination conditions are met, such as selecting the `STOP` action or exceeding a predefined step limit. The agent operates in a continuous environment setting and is not restricted to a navigation graph [105].

Benchmark Definition: CoIN-Bench. We introduce CoIN-Bench, a curated benchmark of complex scenarios involving multiple visually similar instances, where fine-grained interaction is essential for target disambiguation. Notably, CoIN-Bench supports both human evaluation and simulated agent-user interactions, and includes a new performance metric that accounts for agent-user interactions.

We build our dataset on top of the large-scale GOAT-Bench [55], which offers a wide range of scenarios sourced from HM3DSem [97] and rendered using Habitat Simulator [105]. GOAT-Bench provides instance references in multiple formats, such as category labels and natural language descriptions, making it a strong foundation for our work. GOAT-Bench includes a `Train` split intended for training navigation policies, along with three evaluation splits: `Val Seen`, `Val Seen Synonyms`, and `Val Unseen`. Specifically, `Val Seen` contains objects present in the training split, `Val Seen Synonyms` introduces object names that are synonymous with those in `Train`, while `Val Unseen` features entirely new object categories not encountered during training. Since GOAT-Bench's `Train` split is reserved for training, we design CoIN-Bench specifically as an evaluation benchmark. To ensure fair comparison with models trained on GOAT-Bench, we thus sample episodes from its evaluation splits, *i.e.*, `Val Seen`, `Val Seen Synonyms`, and `Val Unseen`. Since CoIN is designed for settings involving multiple instances of the same object category (*distractors*), we apply a filtering step to retain only episodes with at least $d_{min} = 2$ distractors.

Table 4.4 CoIN-Bench statistics: Average (standard deviation) number of distractors, geodesic distance to the goal, and number of episodes per split.

Statistics	Val Seen	Val Seen Synonyms	*Val Unseen*
Avg. (std) number of distractors	4.58 (1.93)	6.01 (1.96)	5.15 (1.51)
Avg. (std) length (Geodesic)	9.32 (3.43)	9.13 (3.14)	9.86 (3.73)
Avg. (std) length (Euclidean)	7.48 (2.88)	7.50 (2.75)	7.78 (3.39)
Number of Episodes	831	359	459

After filtering, we use the Habitat simulator [105] to assign random start positions to the agent, maintaining a geodesic distance between 5 and 20 meters from the target to introduce variation in navigation difficulty. Given that visual observations are rendered in 3D and their quality depends on scene reconstruction, we manually inspect and remove episodes where the target object has poor resolution, minimal visibility, or is visually indistinguishable from surrounding distractors. We also exclude episodes that require crossing floors to reach the target, following standard protocol. CoIN-Bench consists of 831 episodes from `Val Seen`, 359 from `Val Seen Synonyms`, and 459 from `Val Unseen`, totaling $1,649$ evaluation episodes, comparable in scale to established benchmarks [58, 55, 11]. As summarized in Tab. 4.4, the dataset features an average of approximately five distractors per episode and a mean path length exceeding 7 meters, making it a challenging multi-instance evaluation benchmark. CoIN-Bench's samples are shown in Fig. 4.7.

Evaluation protocol. CoIN-Bench supports evaluation with both:

(i) *real human users*, to assess the strengths and limitations of authentic agent-human interactions

(ii) *simulated user-agent* interactions, enabling scalable, extensive and reproducible experiments.

Simulating these interactions is particularly challenging due to: *(i)* the agent's ability to ask open-ended, template-free questions about any attribute of the target, making it infeasible to predefine an exhaustive question-answer set; and *(ii)* the vast space of possible queries in a continuous 3D environment [105]. To address these challenges, we simulate user responses using a vision-language model (VLM) with access to a high-resolution (1024×1024) image of the target instance in each episode. This

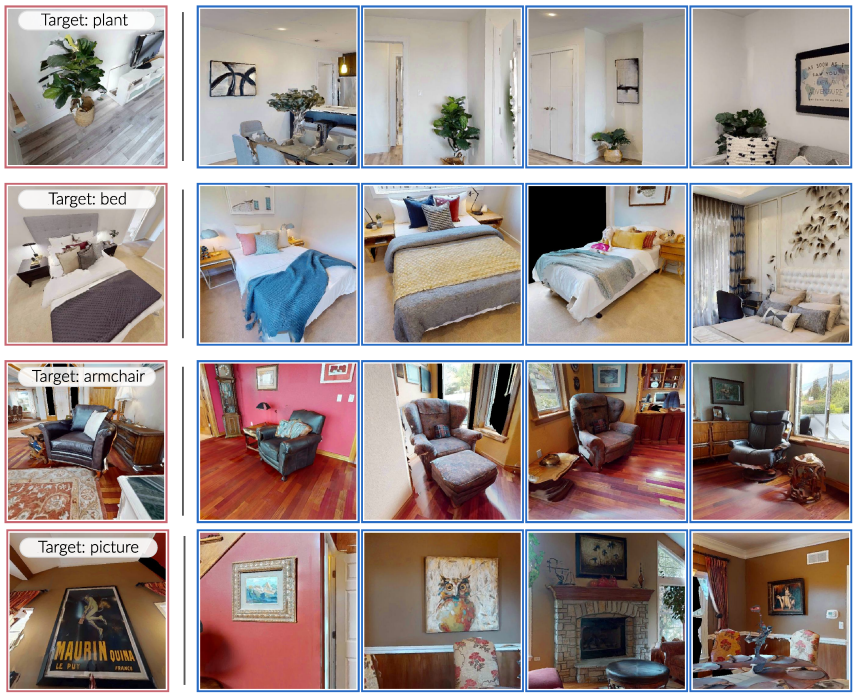


Fig. 4.7 CoIN-Bench can be very challenging when only given the instance category to the agent. We highlight the target instance with red borders, while the distractor instances that exist in the same scene are marked with blue borders.

approach offers richer and more grounded responses compared to relying solely on text-based descriptions [23], as the detailed visual input enables the model to generate more diverse and contextually accurate answers to the agent's queries. An illustration of the evaluation setup, both with real and simulated users, is shown in Fig. 4.8.

4.2.3 Method

Our proposed Agent-user Interaction with UncerTainty Awareness (**AIUTA**), a reasoning module that enriches the agent, is illustrated in Fig. 4.9.

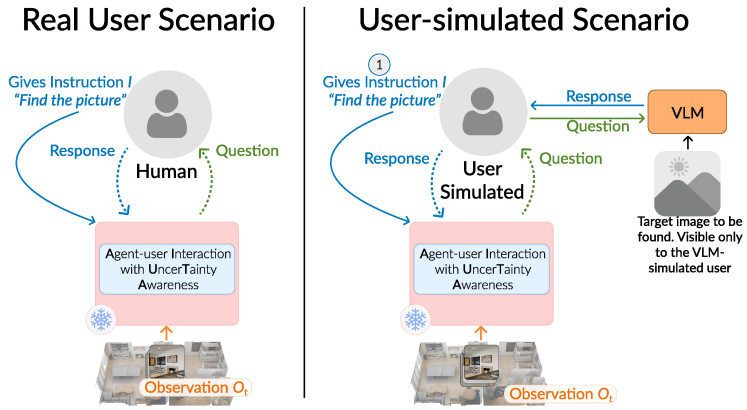


Fig. 4.8 CoIN-Bench evaluation setup. (Left) Real human responding to the agent's question. (Right) Simulated user-agent interactions, where the user responses are provided by a VLM with access to a high-resolution target instance image for scalable and reproducible experimentation.

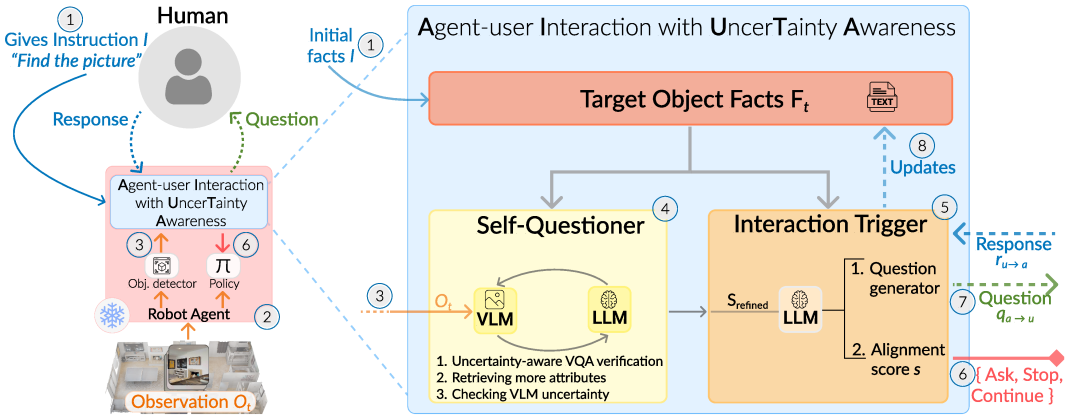


Fig. 4.9 Graphical depiction of **AIUTA**: left shows its interaction cycle with the user, and right provides an exploded view of our method. ① The agent receives an initial instruction I: "Find a c =<object category>". ② At each timestep t, a zero-shot policy π [141], comprising a frozen object detection module [74], selects the optimal action a_t. ③ Upon detection, the agent performs the proposed AIUTA. Specifically, ④ the agent first obtains an initial scene description of observation O_t from a VLM. Then, a ***Self-Questioner*** module leverages an LLM to automatically generate attribute-specific questions to the VLM, acquiring more information and refining the scene description with reduced attribute-level uncertainty, producing $S_{refined}$. ⑤ The ***Interaction Trigger*** module then evaluates $S_{refined}$ against the "*facts*" related to the target, to determine whether to terminate the navigation (if the agent believes it has located the target object ⑥), or to pose *template-free, natural-language* questions to a human ⑦, updating the "*facts*" based on the response ⑧.

Upon receiving the initial user instruction I, such as *"Find the picture"* (① in Fig. 4.9), AIUTA initializes the set of known target facts as $F_{t=0} = \{I\}$. It then activates a zero-shot navigation policy, VLFM [141], which processes the current observation O_t and selects the next action a_t (②). VLFM builds an occupancy map to identify unexplored frontiers and constructs a value map that ranks them by their semantic relevance to the target, using the BLIP-2 [65] vision-language model.

AIUTA is triggered when an object belonging to the target class is detected (③); then, it executes two key components in sequence:

(i) the *Self-Questioner* uses a vision-language model (VLM) and a large language model (LLM) to perform a self-questioning procedure, aiming to derive a detailed and grounded (*i.e.*, hallucinations-free) understanding of the detected object (④). This enables a reliable comparison with the known target facts.

(ii) the *Interaction Trigger* determines whether an interaction with the user is needed by evaluating the observed object description against F_t. Based on this reasoning, the agent either asks a clarification question (`ASK`), halts the episode (`STOP`), or continues navigation (⑤,⑥).

If the agent chooses to `ASK` (⑦), the resulting user response is used to update the fact set F_t (⑧). The navigation ends once the agent determines that the correct target has been found. In the following, the *Self-Questioner* (4.2.3.1) and the *Interaction-Trigger* (4.2.3.2) modules will be fully detailed. The full algorithm is available in *Supp. Mat.*, Section A.2.

4.2.3.1 Self-Questioner

Upon detecting a potential target, the *Self-Questioner* module aims to generate a reliable and detailed description of the observed object. Prior work has shown that generative vision-language models (VLMs) may produce outputs that are only loosely grounded in the visual input, often resulting in hallucinated or inaccurate content [127, 73, 95].

To address this limitation, we incorporate a large language model (LLM) to automatically generate attribute-specific questions for the VLM. Central to our method

is a novel uncertainty estimation technique that enhances the quality of the object description by identifying and filtering out unreliable attributes. The process consists of three key steps: *(i)* generating an initial, attribute-rich description of the detected object based on the visual input; *(ii)* estimating uncertainty in the VLM's responses to validate the reliability of the detected attributes; *(iii)* refining the description by removing attributes deemed uncertain. Each step is described in detail below.

Generation of the initial detection description. The agent begins by prompting the VLM to generate an initial description S_{init} of the current observation O_t, using the prompt:

$$P_{init} = \text{“}\textit{Describe the <target\_object> in the provided image.}\text{”} \tag{4.3}$$

Formally, the response is defined as:

$$S_{init} = \text{VLM}(O_t, P_{init}). \tag{4.4}$$

However, this initial description may omit critical details necessary to identify the specific target instance. For example, when searching for a picture, the content depicted within the picture may not be explicitly mentioned. To address this, we leverage an LLM to generate a set of follow-up *self*-questions $Q_{a\to a}^{details} = \{q_j\}$, aimed at enriching the initial description. These questions are generated based on S_{init} and the current set of known facts F_t, as follows:

$$Q_{a\to a}^{details} = \text{LLM}(P_{details}, S_{init}, F_t) \tag{4.5}$$

where $P_{details}$ is a prompt instructing the LLM to produce detailed, attribute-specific questions (see *Supp. Mat.* Section A.1.2). Each question $q_j \in Q_{a\to a}^{details}$ is then answered by the VLM using the same observation O_t:

$$r_j = \text{VLM}(O_t, q_j). \tag{4.6}$$

All responses $\{r_j\}$ are then concatenated with the initial description S_{init} to produce the enriched detection description:

$$S_{enriched} = S_{init} \cup \{r_j\} \tag{4.7}$$

Perception uncertainty estimation. VLMs can generate hallucinated or inaccurate content [127, 73, 95], which can negatively impact the performance of AIUTA. To address this, we propose a novel training-free technique for estimating the perception uncertainty of VLMs. Directly evaluating this uncertainty is challenging and often requires changes to the model architecture. Instead, we introduce a prompt-guided method based on Shannon entropy for post-hoc uncertainty estimation.

Our goal is to estimate the uncertainty $u \in [0,1]$ of a VLM when answering a specific question q given an observation O_t, such that:

$$(r,u) = \text{VLM}(O_t, q) \tag{4.8}$$

where r is the response and u the uncertainty score. Following [73], we consider an auto-regressive VLM, where $\mathbf{X}_I$ are the image tokens, $\mathbf{X}_P$ are the prompt tokens, and $\mathbf{X}_H$ are the previously generated tokens. The VLM generates a conditional probability distribution p over the vocabulary $\mathbf{y} \in \mathbb{R}^w$:

$$\begin{aligned} \mathbf{y} &\sim p_{\text{VLM}}(\mathbf{y} \mid \mathbf{X}_I, \mathbf{X}_P, \mathbf{X}_H) \\ &\propto \text{softmax}\left(\text{logit}_{\text{VLM}}(\mathbf{y} \mid \mathbf{X}_I, \mathbf{X}_P, \mathbf{X}_H)\right). \end{aligned} \tag{4.9}$$

Since VLM has an unbounded output space and its output probability distribution is over a (large) vocabulary of size w, directly estimating entropy is non-trivial. To address this, since VLMs are typically instruction-tuned [68], we restrict the output space using templated prompts. Specifically, we use:

"<Question>? You must answer with `Yes, No, or ?=I don't know`*."*

This formulation simplifies the problem by: *(i)* reducing the output to a single token (removing the need for length normalization), and *(ii)* bounding the vocabulary size to $w = 3$. We then compute Shannon entropy [107] H of the probability distribution p over vocabulary size w:

$$H(p_{\text{VLM}}) = -\sum_{i=1}^{w} p(y_i) \log p(y_i) \tag{4.10}$$

and normalize it in the range $[0,1]$ as:

$$u = \frac{H}{H_{\max}}, \quad \text{where } H_{\max} = \log(w) = \log(3) \tag{4.11}$$

Given a threshold τ, the certainty level function $C(u,\tau)$ is determined as:

$$C(u,\tau) = \begin{cases} \texttt{Certain}, & u \leq \tau \\ \texttt{Uncertain}, & u > \tau \end{cases} \tag{4.12}$$

To reduce false positives, we also check the object detection with a verification prompt (see *Supp. Mat.* Section A.1.3):

> *"Does the image contain a <target object>? Answer with Yes, No or ?=I don't know."*

This yields the response and uncertainty $(r_{\text{check}}, u_{\text{check}}) = \text{VLM}(O_t, P_{\text{check}})$. According to Eq. 4.12, the AIUTA pipeline proceeds only if $r_{\text{check}} =$ *"Yes"* and $u_{\text{check}} = \texttt{Certain}$; otherwise, the agent continues exploring.

To further remove uncertain attributes, we prompt the LLM to extract a set of attribute-value pairs from the enriched description $S_{enriched}$:

$$K_t = \{(k_j, v_j)\} \tag{4.13}$$

e.g., ("frame", "black"), ("content", "RGB image of a family"), etc. For each attribute k_j, we prompt the LLM to generate a list of J questions:

$$Q_{a\to a}^{\text{attribute}} = \{q_j\}_{j=1}^{J} \tag{4.14}$$

to be self-answered by the agent. These questions are produced via:

$$Q_{a\to a}^{\text{attribute}} = \text{LLM}(P_{\text{self questions}}, F, S_{enriched}) \tag{4.15}$$

where $P_{\text{self questions}}$ is the LLM prompt detailed in *Supp. Mat.* (Section A.1.4). Then, each question q_j is answered by the on-board VLM, yielding:

$$(r_j, u_j) = \text{VLM}(O_t, q_j) \tag{4.16}$$

This process confirms or discards attributes based on VLM uncertainty, resulting in a final, refined object description $S_{refined}$.

Detection Description Refinement. To produce the final refined object description $S_{refined}$, we prompt the LLM to filter out uncertain attributes from the enriched description $S_{enriched}$. This decision is guided by the set of self-questions, corresponding VLM responses, and associated uncertainty scores, denoted as $\{q_j, r_j, u_j\}$. Formally, the refinement process is defined as:

$$S_{refined} = \text{LLM}(P_{\text{refined}}, \{q_j, r_j, u_j\}, S_{enriched}) \tag{4.17}$$

where P_{refined} is the instruction prompt provided to the LLM (see *Supp. Mat.* Section A.1.5).

4.2.3.2 Interaction Trigger

Using the refined and accurate object description $S_{refined}$, the *Interaction Trigger* queries an LLM to determine whether to engage the user or continue navigation autonomously. Specifically, we prompt the LLM to compute a similarity score s that reflects the alignment between the scene description $S_{refined}$ and the current set of target object facts F_t. Formally, the similarity score s is estimated as:

$$s = \text{LLM}(P_{\text{score}}, S_{refined}, F_t) \tag{4.18}$$

where P_{score} is a prompt that instructs the LLM to evaluate semantic similarity (see *Supp. Mat.* Section A.1.6). The agent selects an action based on the similarity score s as follows:

(i) If $s \geq \tau_{\text{stop}}$, the agent concludes that the target has been found and terminates the episode (`STOP`).

(ii) If $s < \tau_{\text{skip}}$, the detected object is considered unrelated to the known target facts, and the agent continues exploring without querying the user.

(iii) If $\tau_{\text{skip}} \leq s < \tau_{\text{stop}}$, the detection is partially aligned with the target facts, prompting the agent to ask a clarifying question to reduce uncertainty (`ASK`).

When selecting the `ASK` action, the agent leverages the LLM to generate an informative question $q_{a \to u}$ aimed at maximizing information gain. This question is conditioned on both the current facts F_t and the refined detection description $S_{refined}$.

To reduce the number of LLM calls, the question generation is incorporated directly within the P_{score} prompt. Upon receiving the human response $r_{u \to a}$, the agent updates its knowledge base:

$$F_t \leftarrow F_t \cup r_{u \to a} \tag{4.19}$$

This update improves the grounding of future decisions and enhances the effectiveness of subsequent agent-user interactions.

4.2.4 Experimental Results

Metrics. We consider an episode successful if the agent selects the action `STOP` within 0.25m of the target viewpoints. If not located, the exploration ends after 500 navigation steps. Following embodied navigation standards, we use the following metrics: Success Rate, `SR` (↑), our primary metric (in gray), and Success rate weighted by Path Length, `SPL` (↑). Additionally, we introduce the *average Number of Questions asked*, `NQ` (↓) in successful episodes to measure the amount of user input.

Implementation Details. We use LLaVA-NeXT [69] (LLaVA 1.6 with Mistral LLM 7B) as the VLM and GPT-4o [48] as the LLM. The user interaction number is limited to a maximum of 4 rounds per detected object. We empirically set $\tau = 0.75$ (Eq. 4.12), $\tau_{stop} = 7$ and $\tau_{skip} = 5$ as they yield the best result.

Baselines. We compare AIUTA against state-of-the-art methods for `ObjectNav` and `InstanceObjectNav`: the SenseAct-NN Monolithic Policy (Monolithic) [55], PSL [112], OVON [142], and the zero-shot, training-free VLFM [141].

To highlight the difficulty of our proposed CoIN-Bench dataset, we include two models trained on GOAT-Bench [55]: Monolithic and OVON. These methods are evaluated on the "*Seen*" splits of GOAT-Bench, where target categories are present during training (Section 4.2.2). PSL, in contrast, is trained on the ImageNav task and transferred to the language-driven Instance Navigation task. Notably, both Monolithic and PSL require a fully detailed textual description d of the target instance. OVON and VLFM, on the other hand, use only the target category c as input. Tab. 4.5 summarizes the input formats and training conditions for each method on CoIN-Bench, following the same evaluation splits used in [55]: `Val Seen`, `Val`

`Seen Synonym`, and `Val Unseen`.

Results with simulated user-agent interaction. As shown in Tab. 4.5, training-based methods achieve stronger performance on `Val Seen` and `Val Seen Synonyms` compared to `Val Unseen`, indicating poor generalization to novel object categories. This trend is especially evident in policies trained on GOAT-Bench (marked with †), where performance drops sharply. For example, OVON's `SR` falls from 15.88 to 2.61, and Monolithic's from 13.09 to 0.22.

In contrast, AIUTA, despite being training-free, outperforms training-based methods on `Val Unseen` and maintains consistently strong performance across all splits. Interestingly, while OVON outperforms AIUTA on `Val Seen Synonyms`, our approach surpasses PSL and Monolithic in both `SR` and `SPL`. This is notable given that PSL and Monolithic operate with detailed target descriptions, while AIUTA only takes the category name as input. A possible explanation lies in the limited ability of CLIP-based models to encode fine-grained instance-level descriptions, as observed in prior works [55, 32]. Furthermore, when comparing results to those reported on GOAT-Bench, the performance degradation of baselines (*e.g.*, Monolithic) on CoIN-Bench demonstrates the increased difficulty introduced by multi-instance ambiguity. Notably, VLFM [141], which also receives only the target category as

Table 4.5 CoIN-Bench is challenging. AIUTA, while being *training-free*, achieves strong performance by outperforming trained policies (top rows) and significantly surpassing the zero-shot VLFM, across *all* splits, through effective user interaction. In contrast, policies trained on GOAT-Bench (denoted with †), the foundation of CoIN-Bench, fail to generalize to novel categories (Val Unseen). We report the `SR` (main metric, in **bold** *w.r.t* training free-methods), `SPL`, and the number of questions `NQ`. Input types: *c* for object category, *d* for its description.

Method	Model Condition		Val Seen			Val Seen Synonyms			*Val Unseen*		
	Input	Training-free	SR ↑	SPL ↑	NQ ↓	SR ↑	SPL ↑	NQ ↓	SR ↑	SPL ↑	NQ ↓
Monolithic† [55] (*CVPR-24*)	d	✗	6.62†	3.11	-	13.09†	6.45	-	0.22†	0.05	-
PSL [112] (*ECCV-24*)	d	✗	8.78	3.30	-	8.91	2.83	-	4.58	1.39	-
OVON† [142] (*IROS-24*)	c	✗	8.18†	5.24	-	15.88†	11.35	-	2.61†	1.29	-
VFLM [141] (*ICRA-24*)	c	✔	0.36	0.28	-	0.00	0.00	-	0.00	0.00	-
AIUTA (*ours*)	c	✔	**7.42**	2.92	1.67	**14.38**	7.99	1.36	**6.67**	2.30	1.13

input, fails nearly all episodes across all splits, with `SR` approaching zero. This is expected, given the high number of distractors in each scene (Tab. 4.4) and the lack of instance-level discrimination capabilities in ObjectNav methods.

In contrast, AIUTA, built on top of VLFM but enhanced with instance-specific interaction-driven reasoning, successfully identifies the correct target instance through minimal user interaction (NQ$<$ 2 across all splits). This leads to a substantial performance boost, achieving an approximately 14$\times$ improvement on Val Seen Synonyms, and around a 7$\times$ improvement on both Val Seen and Val Unseen.

AIUTA's question diversity. To illustrate the diversity of questions to the user generated by AIUTA, we collect 414 question samples made by the agent, compute embedding using Sentece-Bert [102] and visualize them using UMAP [81] for dimensionality reduction. The results, shown in Fig. 4.10, demonstrate that AIUTA generates questions covering a wide range of attributes, such as color, material, style, and spatial arrangement.

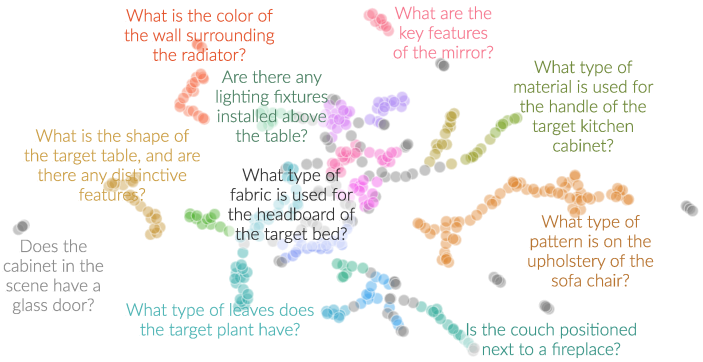


Fig. 4.10 AIUTA generates questions covering a wide range of attributes, such as color, material, style, and spatial arrangement.

Is simulated user-agent interaction reliable? To assess the reliability of our simulated user-agent interaction setup, we additionally conduct a human evaluation on a subset of CoIN-Bench. We randomly sample 40 episodes, each featuring a *detectable* target instance, from all evaluation splits, to reduce participant workload and ensure meaningful interactions. As expected, the SR for this subset is higher than the averages reported in Tab. 4.5.

A total of 20 participants, varying in age and background, each interact with the agent in two episodes using a chat-based interface. Participants are shown an image of the target instance and initiate the task using the fixed prompt "Find the <category>" before responding to the agent's natural language questions. Results

comparing real human interactions and simulated ones are reported in Tab. 4.6. We find no significant differences in key metrics, indicating that our simulation setup offers a *faithful and reproducible* proxy for evaluating user-agent interactions.

Table 4.6 Real human *vs* simulated user-agent interaction.

User type	CoIN-Bench subset		
	SR ↑	SPL ↑	NQ ↓
Simulated	42.50	15.48	1.10
Real Human	42.50	17.44	1.29

Table 4.7 Ablation of AIUTA components on the CoIN-Bench `Train` split.

Self-Questioner	Skip-Question	Ablation split		
		SR ↑	SPL ↑	NQ ↓
✗	✗	9.21	5.86	3.57
✗	✔	8.55	4.84	2.69
✔	✗	9.87	6.50	4.60
✔	✔	**14.47**	**7.22**	**1.68**

Ablation I: Impact of Individual Components in AIUTA. We introduce the *Ablation* split, derived from the GOAT-Bench `Train` set, using the same filtering procedure detailed in Section 4.2.2. We select `Train` due to its broader semantic category coverage. Since AIUTA is *training-free*, using this data for evaluation remains a fair choice. Tab. 4.7 demonstrates the impact of the *Self-Questioner* and the *Skip-Question* mechanism within the Interaction Trigger. When both components are disabled (row 1), `SR` drops to 9.21%, and the average number of questions (`NQ`) is high. Removing only the Self-Questioner (row 2) results in a performance drop and fewer questions, as expected. Enabling only the Self-Questioner (row 3) slightly improves `SR` to 9.87%, but keeps `NQ` high. Finally, activating both modules (row 4) yields the highest `SR` of 14.47% and reduces `NQ` to 1.68, confirming that both components are essential for achieving strong and efficient performance.

Ablation II: VLM uncertainty estimation on IDKVQA. VLM uncertainty estimation is a key component of the Self-Questioner module, enabling the agent to reduce hallucinations and improve answer reliability. To evaluate our proposed technique, we introduce IDKVQA, a dedicated VQA dataset comprising 502 questions over 102 images sampled from GOAT-Bench [55]. Notably, each question is annotated by three human annotators selecting from `{Yes, No, I Don't Know}`, allowing models to abstain when visual information is insufficient. Samples from IDKVQA are shown in Fig. 4.11.

We benchmark our *Normalized-Entropy*-based uncertainty estimation method against three recent methods:

Table 4.8 Results of different selection functions and their corresponding *Effective Reliability* rate $\Phi_{c=1}$ [135] on the IDKVQA dataset.

VLM Model	Selection Function	$\Phi_{c=1}$
LLaVA llava-v1.6-mistral-7b-hf	MaxProb	15.94
	LP [148]	14.01
	Energy Score [75]	20.45
	Normalized Entropy (ours)	**21.12**

(i) *MaxProb*, which selects the answer with the highest predicted probability. It does not incorporate uncertainty estimation.;

(ii) *LP* [148], a logistic regression model trained as a linear probe on the logits of the first generated token. The model is trained on the *Answerable*/*Unanswerable* classification task using the VizWiz VQA dataset [38], which includes 23,954 images for training. When applied to IDKVQA, the logistic regression model first predicts whether the question q is *Answerable* or *Unanswerable*. If the question is deemed answerable, the response r with the highest probability is selected among {`Yes`, `No`}; otherwise, the response `I don't know` is returned.

(iii) *Energy Score* [75], a method for out-of-distribution detection based on an energy score-based framework. An energy score is computed to identify whether the given question-image pair is OOD. If the pair is classified as OOD, the response `I don't know` is returned; otherwise, the response with the highest probability is selected among {`Yes`, `No`}.

Performance is reported in Tab. 4.8 using the *Effective Reliability* metric Φ_c from [135], which jointly measures VQA accuracy and the ability to abstain when uncertain. Our approach achieves the highest $\Phi_{c=1}$ score of 21.12, validating the effectiveness of our entropy-based uncertainty estimation.

***Ablation III*: Sensitivity analysis of the threshold τ.** We analyze the sensitivity of the threshold parameter τ (as defined in Eq. 4.12) for our *Normalized-Entropy*-based uncertainty estimation and the second-best performing method, Energy Score [75]. To this end, we subsample the dataset to 50%, 70%, and 100% of its original size. Specifically, we create five sets containing 50% of the question-answer pairs from CoIN-Bench, five sets comprising 70% of the question-answer pairs, and also use

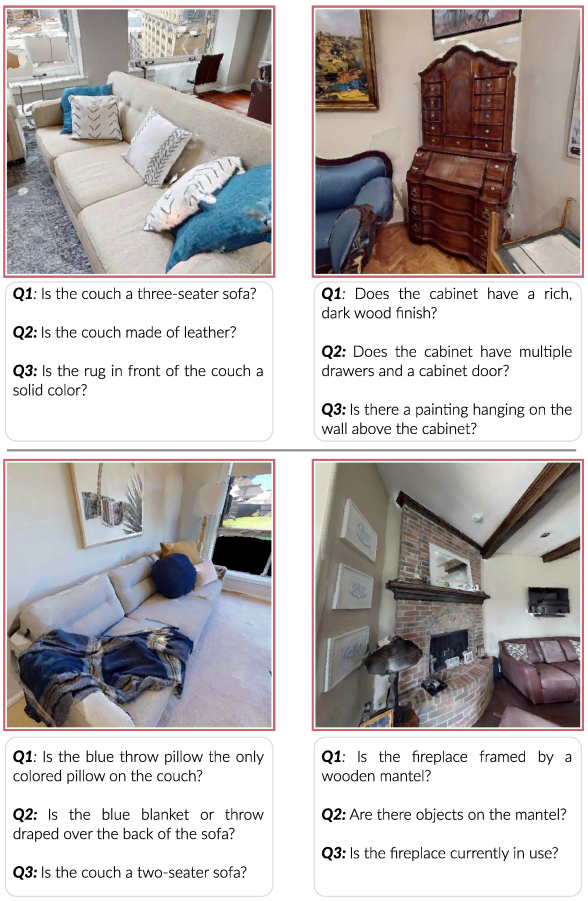


Fig. 4.11 Examples from IDKVQA, showing images and the questions generated by the LLM.

the full dataset (100%) for a total of 11 datasets. For each subsample, we determine the optimal threshold τ^* and assess robustness by evaluating the *Effective Reliability* metric $\Phi_{c=1}$ over 30 thresholds sampled around τ^* and normalized in the $[0,1]$ range. Our goal is to analyze how $\Phi_{c=1}$ changes across these neighborhoods: if the values are spread out, it means that the method is very sensitive to small changes of τ near the optimal value, whereas if they are more tightly distributed it means that it is more robust. As shown in Fig. 4.12, our method yields a narrower inter-quartile range and a more concentrated distribution of $\Phi_{c=1}$ scores, while Energy Score [75] suffers from greater performance degradation as the threshold deviates from τ^*. This effect becomes more pronounced as the dataset size decreases, highlighting the improved stability of our approach under limited data. In addition, Energy Score relies on unbounded logit values, which complicates threshold calibration. In contrast, our

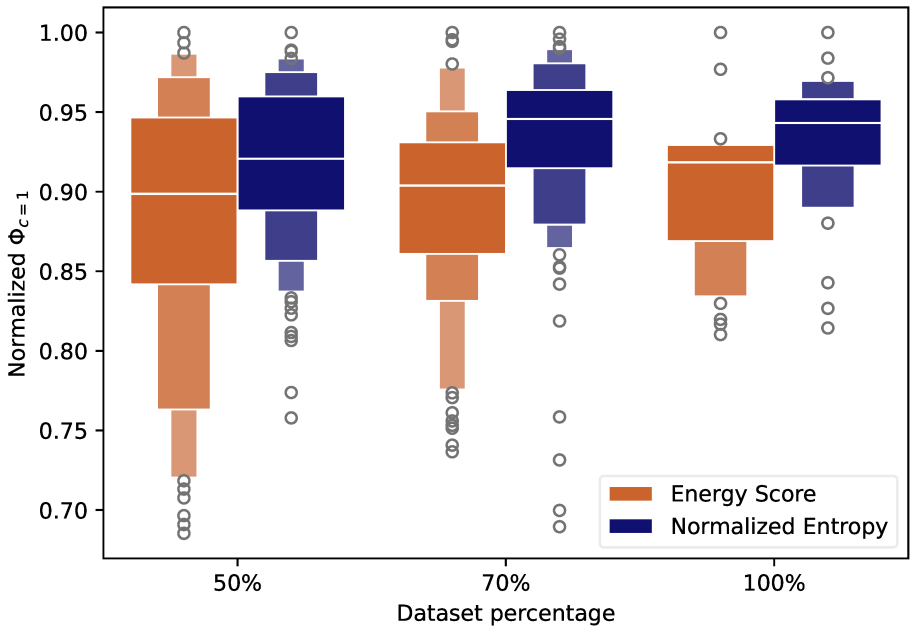


Fig. 4.12 τ sensitivity results. For each method, 30 new τ values are sampled symmetrically around the optimal threshold τ^*. The x-axis shows the set size as a percentage of the original IDKVQA dataset size, while the y-axis displays the normalized ER $\Phi_{c=1}$.

uncertainty values are normalized in $[0,1]$, making the selection of an optimal τ more efficient.

4.2.5 Conclusion

We introduced the Collaborative Instance object Navigation task, in which the agent collaborates with the user during navigation to resolve uncertainties about the target instance. Through extensive experiments, we show that existing trained methods fail to generalize to unseen categories, while our training-free AIUTA, leveraging a novel self-dialogue mechanism and uncertainty estimation, achieves strong performance across all validation splits. Moreover, our simulated user-agent interaction closely aligns with human evaluations, enabling scalable and reproducible experimentation. Future work will explore model optimization for embodied deployment, aiming to reduce inference costs, and extend the interaction scope to include action-level instructions.

Chapter 5

Conclusions

5.1 Summary of the main contributions

Although embodied AI agents have shown remarkable performance on benchmark tasks, they are often evaluated under idealized conditions, assuming perfect perception and flawless communication, assumptions that rarely hold in practice. To bridge this gap, this thesis addressed two complementary sources of uncertainty that undermine agent robustness: *internal uncertainty* arising from imperfect sensors, and *external uncertainty* originating from ambiguous or erroneous human instructions. We now summarize our main contributions along these two axes.

Sensor Robustness: Object Detection. We introduced POMP-BE-PD [116], a training-free POMCP-based planner for active visual search that explicitly models object detector unreliability. During exploration, the agent maintains and updates a probability distribution over candidate object locations by incorporating detector statistics. A Bayesian inference mechanism over this distribution enables the agent to reduce false positive errors by 32%. Additionally, improved belief update and docking procedures enhance the agent's robustness and reduce navigation time. Our method outperforms state-of-the-art baselines in both success rate and path efficiency, particularly in challenging environments.

Instruction Robustness: Erroneous Instructions. Having addressed internal uncertainty, we now turn to the external source: imperfect human instructions. We introduced `R2RIE-CE`, a new benchmark for `VLN-CE` where various types of errors

are systematically injected into navigation instructions. To address this setting, we propose IEDL [118], a model capable of identifying erroneous instructions and localizing the errors with sub-sentence precision. We further extended this method with I2EDL [117], which performs *online* instruction error detection and localization during navigation. Together, our benchmark, task formulations, and methods represent significant steps toward more robust and adaptive `VLN-CE` agents capable of operating under real-world linguistic imperfections.

Instruction Robustness: Ambiguous Instructions. We introduced the CoIN task, in which the agent collaborates with the user during navigation to resolve uncertainties about the target object instance. To address this challenge, we proposed AIUTA [119], a training-free module for human–agent interaction reasoning that leverages Vision-and-Language Models (VLMs) and Large Language Models (LLMs). Upon object detection, a Self-Questioner component initiates an internal dialogue to generate a complete and accurate description of the agent's observations, while a novel uncertainty estimation technique removes perceptual hallucinations. Subsequently, an Interaction Trigger module decides whether to query the user, proceed with navigation, or halt exploration, thereby minimizing unnecessary user input. Experiments on our proposed CoIN-Bench demonstrate that AIUTA serves as a strong training-free baseline, while existing language-driven instance navigation methods fail to generalize to complex multi-instance scenarios.

5.2 Limitations & Future Work

While the contributions above represent significant steps toward robust embodied agents, several limitations remain. We now discuss these limitations and outline directions for future research, organized according to the same structure as our contributions.

Sensor Robustness: Object Detection. While POMP-BE-PD introduces an effective framework for Active Visual Search (AVS), several limitations remain. First, the method assumes access to an accurate 2D floor map of the environment. In real-world scenarios, such information may be unavailable or incomplete. Although POMCP-based planners for unknown environments have been proposed [35], they do not account for uncertainty-aware planning as done in our framework. Future work

could explore integrating online mapping or SLAM-based methods to incrementally build a map during exploration while maintaining probabilistic reasoning under detection uncertainty.

Second, despite incorporating detection uncertainty, the method relies on precomputed detector statistics, such as category-wise precision and recall. This dependence limits adaptability to new detectors or environmental conditions when such statistics are not known a priori. A promising direction would be to develop adaptive mechanisms for estimating or learning these statistics online, allowing the system to operate effectively in unseen domains. Additionally, incorporating online learning strategies and multi-view selection policies (*i.e.*, if uncertain, let the agent select the object from another point of view) could help reduce target uncertainty during navigation.

Third, the probability map used in this work is designed for single-object search, and does not consider the spatial relationship between objects to guide planning. Extending the framework to support multi-object search would require algorithmic modifications and may increase computational demands. Future extensions could involve hierarchical planning techniques that reason jointly over multiple object instances while preserving computational efficiency. Additionally, an interesting area of research could be exploring how to integrate LLM common sense into the POMDP planning procedure.

Finally, all evaluations are performed in simulated environments using the Active Vision Dataset Benchmark (AVDB). The lack of real-world deployment leaves open questions regarding the robustness of the approach in the presence of additional sensor-related issues not considered in this work.

Instruction Robustness: Erroneous Instructions. While the proposed framework improves the robustness of Vision-and-Language Navigation (VLN) in continuous environments by detecting and localizing instruction errors, it presents several limitations.

First, the `R2RIE-CE` benchmark is constructed by artificially injecting errors into existing VLN datasets (*i.e.*, `R2R-CE`), rather than collecting human-generated erroneous instructions. Although the injected errors consider commonsense relationships, they may not fully capture the complexity and variability of real-world mistakes. Future work could involve collecting real human-in-the-loop navigation data, where

errors occur naturally, to improve the realism and generalizability of instruction error detection models.

Second, the Instruction Error Detector & Localizer (IEDL) operates offline, relying on complete trajectories and full instructions. Although I2EDL demonstrates how a pre-trained IEDL model can be used online for recovery during navigation, it remains unclear how to train an error-aware policy from scratch and how to simulate a human in an interactive setting. Developing online, *end-to-end trainable* models capable of detecting and correcting errors during navigation, without relying on a full trajectory remains an open challenge. Additionally, future research should explore interactive learning paradigms where agents can request clarifications, simulate user feedback, or adaptively learn from mistakes in real time. To this end, understanding the model's uncertainty about when it should ask for information remains an open challenge.

Finally, experiments are confined to simulated environments. Real-world deployment would likely introduce new types of instruction errors and environmental challenges not represented in simulation.

Instruction Robustness: Ambiguous Instructions. CoIN introduces a novel task that, for the first time, supports *online, natural, and template-free* human interaction to resolve target ambiguity during navigation. While AIUTA performs effective embodied reasoning in this setting, this work lays the foundation for a new class of simulated interaction types, paving the way for future research on more directed, flexible, and context-aware agent behaviors, as outlined below.

First, the current setup uses a vision-language model (VLM) to simulate the human user, supporting only natural language interactions. While this avoids hand-crafted templates, it does not fully capture the variability and richness of real human communication, such as non-verbal cues, disfluencies, or multi-modal grounding. Future work should explore human-in-the-loop data collection to better understand the interaction dynamics and improve realism.

Second, the AIUTA framework relies heavily on large language models (LLMs), where performance improves with model size [52]. However, these larger models incur significant inference costs, both computationally and financially, making them impractical for real-time, on-board processing in resource-constrained settings (e.g., mobile robots or drones). Moreover, many LLMs require sending user data or

scene information to external servers for processing, raising concerns about data privacy and the secure handling of sensitive or personally identifiable information for embodied domains. Distilling the reasoning capabilities of large models into smaller, efficient variants remains an open and critical research challenge.

Third, current natural language dialogues in CoIN are restricted to resolving ambiguities about the target object. This excludes other important forms of human-agent interaction, such as navigation-level guidance (*e.g.*, *"go left now"*) or feedback-driven correction (*e.g.*, *"you passed it"*). Expanding dialogue capabilities toward richer forms of interaction could improve accessibility, enable collaboration with diverse users, and enhance task success in complex environments. However, due to our proposed VLM-simulated user setup, which does not rely on templates or offline-computed datasets, simulating such interaction types online remains an open research question.

Fourth, AIUTA's embodied reasoning is triggered upon object detection. While this is convenient, as it allows for a training-free approach, alternative triggering mechanisms could be proposed, such as leveraging model uncertainty. However, understanding model uncertainty, *i.e.*, enabling the agent to raise the question *"I don't have all the information; it would be better to ask"* remains an open research question.

Fifth, AIUTA can collect target object facts (*i.e.*, information about the target object) as more agent–human interactions occur. While the dialogues are grounded in these facts (to maximize the effectiveness of human–agent interaction), this information is currently not used during navigation to improve planning. Future work could extend the use of online feedback into the navigation policy.

Finally, all experiments are conducted in simulated environments. While this allows controlled benchmarking, it does not account for the unpredictability and noise of real-world deployment, such as speech recognition errors, visual sensor limitations, or dynamic scene changes. Future work should explore real-world implementations to test robustness and generalization.

5.3 Embodied AI: The Next Era

Having explored the current limitations and proposed future directions of the work presented, I now step back to consider a broader question:

Where is Embodied AI headed?

This section reflects my perspective on the research frontiers that may shape the field in the coming years, before discussing how the contributions of this thesis position the community toward these goals (Section 5.4).

Embodied maps. Embodied agents today lack a scalable (to large environments), semantic, queryable (*i.e.*, able to perform searches given language or visual features), and updatable map of their environment. In real-world deployment, agents should perform better over time as they gather experience in an environment, as also shown by [59].

Generalist Embodied Agents. Currently, we are still focused on solving one task at time using ad-hoc solutions and architectures (*e.g.*, separate agents for `ObjectNav`, `InstanceObjectNav` and `VLN`). Generalist embodied agents are still in their infancy, even though impressive performance has been achieved [46, 114]. Developing and training such agents remains difficult, particularly from a simulation perspective. As a research community, we should develop shared and unified benchmarks and tasks, modular frameworks, and simulators to accelerate progress in this area.

Embodied Reasoning. As more and more agents are deployed, it becomes essential to develop agents that can reason, mimicking the two distinct modes of cognitive processing (*i.e.*, fast and slow), as introduced by Kahneman [51]. This model offers the benefits of:

- A *slow, System 2* thinking mode for complex problem-solving and analytical tasks, such as *"Is this the object the user is looking for?"* or *"I completed the first part of the instruction; now I should enter the bedroom, since the user mentioned a plant and I'm seeing that."*
- A *fast, System 1* mode for intuitive operations that do not require reasoning.

This "reasoning" procedure is currently being developed especially with large language models [108, 134, 56]. However, embodied reasoning presents additional

challenges. While larger models improve reasoning, they are difficult to deploy on embodied agents due to resource constraints (both from a time and cost perspective). Distillation and new training paradigms are promising directions [108, 24]. Moreover, embodied agents must reason over trajectories, *i.e.*, a history of observations, rather than single steps. Agents should learn *when to reason* and when to act intuitively, dynamically switching between fast and slow modes depending on environmental conditions. Incorporating uncertainty estimates may help agents decide which mode to use under different conditions.

Interpretability and Explainability. As embodied agents enter real environments, transparent decision-making processes become essential for safety and trust. Agents should generate human-readable explanations for their actions, clarifying why they chose a particular path, asked for help, or aborted a task.

Human-Centric. The next generation of agents must engage naturally with humans when they are uncertain. Beyond verbal dialogue, they should support multimodal cues such as gestures, vision-based instructions (*e.g.*, pointing) and handle implicit commands (*e.g.*, "get my cup"). Human demonstrations, especially video data, will play a larger role in training, enabling robots to retarget observed actions into their own motor commands.

Of all these directions, Vision–Language–Action (VLA) models stand out. They combine semantic reasoning from LLMs and perceptual skills from VLMs into a unified representation that powers downstream tasks.

I conclude by highlighting the greatest challenge ahead: integrating rich mapping, generalist VLA capabilities, embodied reasoning, interpretability, and human-centric interaction into compact models (*e.g.*, 3B), efficient enough for real-world deployment. In ten years' time, I look forward to revisiting these predictions and assessing how far we've actually come.

5.4 Potential Impact

Having outlined the research frontiers that may shape Embodied AI in the coming years, I now reflect on how the contributions of this thesis position the field toward these goals and their potential impact on the broader computer vision and embodied AI communities.

Toward Richer Embodied Maps. One promising approach towards a semantic and queryable map is to integrate vision–language aligned features and latent codes into a top-down map. In [115], we show that by projecting vision-language aligned embeddings onto the map, we can support natural-language queries (*e.g.*, *"a couch in front of the windows in the living room"*). Moreover, by projecting latent codes and converting them into Neural Radiance Fields, we show that it is also possible to achieve image synthesis by rendering images from *any* camera pose. This work provides the community with a first step toward unified map representations that support both semantic retrieval and visual rendering capabilities (*e.g.*, previewing the agent's destination, the path taken, or the searched location). Yet, image generation lacks fine-grained detail, and updating such maps and scaling them to large, complex environments with precise localization and synthesis remains an open challenge.

Toward Embodied Reasoning. As discussed in Section 5.3, future agents will need to reason over their observations, switching between fast, intuitive responses and slow, deliberate analysis. The AIUTA framework [119] takes early steps in this direction. Built on top of a fast zero-shot policy, AIUTA's Self-Questioner component initiates an internal dialogue to generate accurate descriptions of the agent's observations, which is crucial for deciding when deeper reasoning or external clarification is required. Furthermore, the uncertainty estimation technique further enables the agent to recognize when its perceptions may be unreliable. These contributions lay the groundwork for future research on *when* and *how* embodied agents should engage in deliberate reasoning versus intuitive action.

Toward Robust Real-World Deployment. A recurring theme throughout this thesis is the gap between idealized benchmark conditions and the complexities of real-world environments. POMP-BE-PD [116] demonstrates that explicitly modeling perception uncertainty leads to more reliable navigation, reducing false positive by 32%. The R2RIE-CE benchmark and the IEDL/I2EDL [118, 117] methods establish a systematic framework for studying instruction errors in Vision-and-Language Navigation. Similarly, the CoIN [119] task and CoIN-Bench address the challenge of ambiguous instructions. Together, these contributions provide the community with new *benchmarks*, *evaluation protocols*, and *methods* that prioritize robustness, a prerequisite for deploying embodied agents in homes, hospitals, warehouses, and other uncontrolled settings.

Toward Human-Centric Agents. The next generation of embodied agents will interact naturally with humans, especially under uncertainty. Unlike prior work that relies on scripted dialogues, the CoIN [119] task and the AIUTA framework represent a foundational step toward this vision, introducing the first benchmark that supports online, natural, and template-free human–agent interaction during navigation. These contributions provide the community with both a task formulation and a strong baseline for studying human-agent collaboration.

Toward Interpretability and Trust. Transparent decision-making is essential for safe deployment. Although interpretability was not the primary focus of this thesis, several contributions support this goal. IEDL [118] localizes errors within instructions with sub-sentence precision, providing interpretable feedback about which parts of an instruction may be unreliable. I2EDL [117] further extends IEDL, triggering in an *online* fashion the user-agent interaction upon the detection of instruction errors during navigation. AIUTA's [119] Self-Questioner generates natural language descriptions of the agent's observations, offering a window into its perceptual understanding. These mechanisms enable users and developers to better understand agent behavior, facilitating error analysis and trust-building.

Finally, I conclude by highlighting the accepted workshop at the International Conference on Intelligent Robots and Systems (IROS 2025),*"Human-Aware Embodied AI"*, which I co-organized. HEAI addressed the challenges of creating seamless human-agent Co-Habitats, focusing on trustworthiness (*when agents don't know*), robustness (*when agents ask for help*), and intuitive collaboration (*bi-directional interaction*).

References

[1] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J. Ballard, Joshua Bambrick, Sebastian W. Bodenstein, David A. Evans, Chia-Chun Hung, Michael O'Neill, David Reiman, Kathryn Tunyasuvunakool, Zachary Wu, Akvilė Žemgulytė, Eirini Arvaniti, Charles Beattie, Ottavia Bertolli, Alex Bridgland, Alexey Cherepanov, Miles Congreve, Alexander I. Cowen-Rivers, Andrew Cowie, Michael Figurnov, Fabian B. Fuchs, Hannah Gladman, Rishub Jain, Yousuf A. Khan, Caroline M. R. Low, Kuba Perlin, Anna Potapenko, Pascal Savy, Sukhdeep Singh, Adrian Stecula, Ashok Thillaisundaram, Catherine Tong, Sergei Yakneen, Ellen D. Zhong, Michal Zielinski, Augustin Žídek, Victor Bapst, Pushmeet Kohli, Max Jaderberg, Demis Hassabis, and John M. Jumper. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 630(8016):493–500, Jun 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-07487-w. URL https://doi.org/10.1038/s41586-024-07487-w.

[2] Christopher Amato and Frans Oliehoek. Scalable Planning and Learning for Multiagent POMDPs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 29(1), February 2015. ISSN 2159-5399. doi: 10.1609/aaai.v29i1.9439. URL http://dx.doi.org/10.1609/aaai.v29i1.9439.

[3] Phil Ammirato, Alexander C. Berg, and Jana Kosecka. Active Vision Dataset Benchmark. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, page 2127–21273. IEEE, June 2018. doi: 10.1109/cvprw.2018.00277. URL http://dx.doi.org/10.1109/cvprw.2018.00277.

[4] Phil Ammirato, Cheng-Yang Fu, Mykhailo Shvets, Jana Kosecka, and Alexander C. Berg. Target Driven Instance Detection, 2019. URL https://arxiv.org/abs/1803.04610.

[5] Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. BEVBert: Multimodal Map Pre-training for Language-guided Navigation. In *ICCV*, 2023. URL https://doi.org/10.48550/arXiv.2212.04385.

[6] Dong An, Hanqing Wang, Wenguan Wang, Zun Wang, Yan Huang, Keji He, and Liang Wang. ETPNav: Evolving Topological Planning for Vision-

Language Navigation in Continuous Environments. *TPAMI*, pages 1–16, 2024. URL 10.1109/TPAMI.2024.3386695.

[7] Peter Anderson, Angel X. Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir R. Zamir. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018.

[8] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sunderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments. In *CVPR*, June 2018. doi: 10.1109/cvpr.2018.00387. URL http://dx.doi.org/10.1109/cvpr.2018.00387.

[9] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL Technical Report, 2025. URL https://arxiv.org/abs/2502.13923.

[10] Luca Barsellotti, Roberto Bigazzi, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Personalized Instance-based Navigation Toward User-Specific Objects in Realistic Environments. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=uKqn1Flsbp.

[11] Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. ObjectNav Revisited: On Evaluation of Embodied Agents Navigating to Objects. *arXiv preprint arXiv:2006.13171*, 2020.

[12] Richard Bellman. A Markovian Decision Process. *Indiana University Mathematics Journal*, 6(4):679–684, 1957. ISSN 0022-2518. doi: 10.1512/iumj.1957.6.56038. URL http://dx.doi.org/10.1512/iumj.1957.6.56038.

[13] Steven Bird and Edward Loper. NLTK: the natural language toolkit. In *ACL*, 2004. doi: 10.3115/1219044.1219075. URL http://dx.doi.org/10.3115/1219044.1219075.

[14] Cameron Browne, Edward Powley, Daniel Whitehouse, Simon Lucas, Peter Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A Survey of Monte Carlo Tree Search Methods. *IEEE Trans. Comp. Intell. AI Games*, 4(1):1–43, 2012. URL 10.1109/TCIAIG.2012.2186810.

[15] Luigi Capogrosso, Federico Girella, Francesco Taioli, Michele Chiara, Muhammad Aqeel, Franco Fummi, Francesco Setti, and Marco Cristani. Diffusion-Based Image Generation for In-Distribution Data Augmentation

in Surface Defect Detection. In *Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications, 2024. doi: 10.5220/0012350400003660. URL http://dx.doi.org/10.5220/0012350400003660.

[16] Alberto Castellini, Georgios Chalkiadakis, and Alessandro Farinelli. Influence of State-Variable Constraints on Partially Observable Monte Carlo Planning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, IJCAI-2019. International Joint Conferences on Artificial Intelligence Organization, August 2019. doi: 10.24963/ijcai.2019/769. URL http://dx.doi.org/10.24963/ijcai.2019/769.

[17] Alberto Castellini, Federico Bianchi, Edoardo Zorzi, Thiago D. Simão, Alessandro Farinelli, and Matthijs T. J. Spaan. Scalable Safe Policy Improvement via Monte Carlo Tree Search. In *Proceedings of the 40th International Conference on Machine Learning (ICML 2023)*, pages 3732–3756. PMLR, 2023. URL https://proceedings.mlr.press/v202/castellini23a.html.

[18] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D Data in Indoor Environments. In *2017 International Conference on 3D Vision (3DV)*, page 667–676. IEEE, October 2017. doi: 10.1109/3dv.2017.00081. URL http://dx.doi.org/10.1109/3dv.2017.00081.

[19] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History Aware Multimodal Transformer for Vision-and-Language Navigation. In *NeurIPS*, volume 34, pages 5834–5847. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/2e5c2cb8d13e8fba78d95211440ba326-Paper.pdf.

[20] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Think Global, Act Local: Dual-Scale Graph Transformer for Vision-and-Language Navigation. In *CVPR*, June 2022. doi: 10.1109/cvpr52688.2022.01604. URL http://dx.doi.org/10.1109/cvpr52688.2022.01604.

[21] Ta-Chung Chi, Minmin Shen, Mihail Eric, Seokhwan Kim, and Dilek Hakkanitur. Just Ask: An Interactive Learning Framework for Vision and Language Navigation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(03):2459–2466, April 2020. ISSN 2159-5399. doi: 10.1609/aaai.v34i03.5627. URL http://dx.doi.org/10.1609/aaai.v34i03.5627.

[22] Rémi Coulom. *Efficient Selectivity and Backup Operators in Monte-Carlo Tree Search*, page 72–83. Springer Berlin Heidelberg, 2007. ISBN 9783540755388. doi: 10.1007/978-3-540-75538-8_7. URL http://dx.doi.org/10.1007/978-3-540-75538-8_7.

[23] Yinpei Dai, Run Peng, Sikai Li, and Joyce Chai. Think, Act, and Ask: Open-World Interactive Personalized Robot Navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3296–3303, 2024. doi: 10.1109/ICRA57147.2024.10610178. URL 10.1109/ICRA57147.2024.10610178.

[24] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, 2025. URL https://arxiv.org/abs/2501.12948.

[25] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, volume 1, page 4171 – 4186, 2019. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85083815650&partnerID=40&md5=4986c6d6076c0c91df84d17216b47216.

[26] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, December 1959. ISSN 0945-3245. doi: 10.1007/bf01386390. URL http://dx.doi.org/10.1007/bf01386390.

[27] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

[28] Kiana Ehsani, Tanmay Gupta, Rose Hendrix, Jordi Salvador, Luca Weihs, Kuo-Hao Zeng, Kunal Pratap Singh, Yejin Kim, Winson Han, Alvaro Herrasti, Ranjay Krishna, Dustin Schwenk, Eli VanderBilt, and Aniruddha Kembhavi. SPOC: Imitating Shortest Paths in Simulation Enables Effective Navigation and Manipulation in the Real World. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 16238–16250. IEEE, June 2024. doi: 10.1109/cvpr52733.2024.01537. URL http://dx.doi.org/10.1109/cvpr52733.2024.01537.

[29] Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. Scene memory transformer for embodied agents in long-horizon tasks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2019. doi: 10.1109/cvpr.2019.00063. URL http://dx.doi.org/10.1109/cvpr.2019.00063.

[30] Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-Follower Models for Vision-and-Language Navigation. In *NeurIPS*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/6a81681a7af700c6385d36577ebec359-Paper.pdf.

[31] Chiara Fulgenzi, Anne Spalanzani, and Christian Laugier. Probabilistic motion planning among moving obstacles following typical motion patterns. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 11-15, 2009, St. Louis, MO, USA*, pages 4027–4033. IEEE, 2009. doi: 10.1109/IROS.2009.5354755. URL https://doi.org/10.1109/IROS.2009.5354755.

[32] 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 *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023. doi: 10.1109/cvpr52729.2023.02219. URL http://dx.doi.org/10.1109/cvpr52729.2023.02219.

[33] Qiaozi Gao, Govind Thattai, Suhaila Shakiah, Xiaofeng Gao, Shreyas Pansare, Vasu Sharma, Gaurav Sukhatme, Hangjie Shi, Bofei Yang,

Desheng Zhang, Lucy Hu, Karthika Arumugam, Shui Hu, Matthew Wen, Dinakar Guthy, Shunan Chung, Rohan Khanna, Osman Ipek, Leslie Ball, Kate Bland, Heather Rocker, Michael Johnston, Reza Ghanadan, Dilek Hakkani-Tur, and Prem Natarajan. Alexa Arena: A User-Centric Interactive Platform for Embodied AI. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 19170–19194. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/3d0758f0b95e19abc68c1c8070d36510-Paper-Datasets_and_Benchmarks.pdf.

[34] Xiaofeng Gao, Qiaozi Gao, Ran Gong, Kaixiang Lin, Govind Thattai, and Gaurav S. Sukhatme. DialFRED: Dialogue-Enabled Agents for Embodied Instruction Following. *IEEE Robotics and Automation Letters*, 7(4):10049–10056, 2022. URL 10.1109/LRA.2022.3193254.

[35] Francesco Giuliari, Alberto Castellini, Riccardo Berra, Alessio Del Bue, Alessandro Farinelli, Marco Cristani, Francesco Setti, and Yiming Wang. POMP++: Pomcp-based Active Visual Search in unknown indoor environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, September 2021. doi: 10.1109/iros51168.2021.9635866. URL http://dx.doi.org/10.1109/iros51168.2021.9635866.

[36] Francesco Giuliari, Geri Skenderi, Marco Cristani, Yiming Wang, and Alessio Del Bue. Spatial Commonsense Graph for Object Localisation in Partial Scenes. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022. doi: 10.1109/cvpr52688.2022.01891. URL http://dx.doi.org/10.1109/cvpr52688.2022.01891.

[37] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, and et al. The Llama 3 Herd of Models, 2024. URL https://arxiv.org/abs/2407.21783.

[38] Danna Gurari, Qing Li, Abigale J. Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P. Bigham. VizWiz Grand Challenge: Answering Visual Questions from Blind People. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, June 2018. doi: 10.1109/cvpr.2018.00380. URL http://dx.doi.org/10.1109/cvpr.2018.00380.

[39] Meera Hahn, Amit Raj, and James M. Rehg. Which way is 'right'?: Uncovering limitations of Vision-and-Language Navigation models. In *AAMAS*, page 2415–2417, Richland, SC, 2023. ISBN 9781450394321. URL https://api.semanticscholar.org/CorpusID:253381693.

[40] Xiaoning Han, Huaping Liu, Fuchun Sun, and Xinyu Zhang. Active Object Detection With Multistep Action Prediction Using Deep Q-Network. *IEEE Transactions on Industrial Informatics*, 15(6):3723–3731, June 2019. ISSN 1941-0050. doi: 10.1109/tii.2019.2890849. URL http://dx.doi.org/10.1109/tii.2019.2890849.

[41] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

[42] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. VLN⟳BERT: A Recurrent Vision-and-Language BERT for Navigation. In *CVPR*, June 2021. doi: 10.1109/cvpr46437.2021.00169. URL http://dx.doi.org/10.1109/cvpr46437.2021.00169.

[43] Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. Bridging the Gap Between Learning in Discrete and Continuous Environments for Vision-and-Language Navigation. In *CVPR*, June 2022. doi: 10.1109/cvpr52688.2022.01500. URL http://dx.doi.org/10.1109/cvpr52688.2022.01500.

[44] Jakub Hort, Jan Laczó, Martin Vyhnálek, Martin Bojar, Jan Bures, and Kamil Vlcek. "Spatial navigation deficit in amnestic mild cognitive impairment". *Proc. of the National Academy of Sciences*, 104(10):4042–4047, February 2007. URL http://dx.doi.org/10.1073/pnas.0611314104.

[45] Haoshuo Huang, Vihan Jain, Harsh Mehta, Alexander Ku, Gabriel Magalhaes, Jason Baldridge, and Eugene Ie. Transferable Representation Learning in Vision-and-Language Navigation. In *ICCV*, October 2019. doi: 10.1109/iccv.2019.00750. URL http://dx.doi.org/10.1109/iccv.2019.00750.

[46] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3D world. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024. doi: https://dl.acm.org/doi/abs/10.5555/3692070.3692890. URL https://dl.acm.org/doi/abs/10.5555/3692070.3692890.

[47] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Tomas Jackson, Noah Brown, Linda Luu, Sergey Levine, Karol Hausman, and brian ichter. Inner Monologue: Embodied Reasoning through Planning with Language Models. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 1769–1782. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/huang23c.html.

[48] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[49] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of Hallucination in Natural Language Generation. *ACM Comput. Surv.*, 55(12), March 2023. ISSN 0360-0300. doi: 10.1145/3571730. URL https://doi.org/10.1145/3571730.

[50] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134, 1998. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(98)00023-X. URL https://www.sciencedirect.com/science/article/pii/S000437029800023X.

[51] Daniel Kahneman. *Thinking, fast and slow*. Penguin, London, 2012. ISBN 9780141033570 0141033576.

[52] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling Laws for Neural Language Models, 2020. URL https://arxiv.org/abs/2001.08361.

[53] Sammie Katt, Frans A. Oliehoek, and Christopher Amato. Learning in POMDPs with Monte Carlo Tree Search. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 1819–1827. PMLR, 2017. URL https://dl.acm.org/doi/10.5555/3305381.3305569.

[54] Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. Simple but Effective: CLIP Embeddings for Embodied AI. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022. doi: 10.1109/cvpr52688.2022.01441. URL http://dx.doi.org/10.1109/cvpr52688.2022.01441.

[55] 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 Computer Vision and Pattern Recognition Conference (CVPR)*, page 16373–16383. IEEE, June 2024. doi: 10.1109/cvpr52733.2024.01549. URL http://dx.doi.org/10.1109/cvpr52733.2024.01549.

[56] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large Language Models are Zero-Shot Reasoners. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2205.11916.

[57] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. *Beyond the Nav-Graph: Vision-and-Language Navigation in Continuous Environments*, page 104–120. Springer International Publishing, 2020. ISBN 9783030586041. doi: 10.1007/978-3-030-58604-1_7. URL http://dx.doi.org/10.1007/978-3-030-58604-1_7.

[58] Jacob Krantz, Stefan Lee, Jitendra Malik, Dhruv Batra, and Devendra Singh Chaplot. Instance-Specific Image Goal Navigation: Training Embodied Agents to Find Object Instances. *arXiv preprint arXiv:2211.15876*, 2022. URL https://doi.org/10.48550/arXiv.2211.15876.

[59] Jacob Krantz, Shurjo Banerjee, Wang Zhu, Jason Corso, Peter Anderson, Stefan Lee, and Jesse Thomason. Iterative Vision-and-Language Navigation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 14921–14930. IEEE, June 2023. doi: 10.1109/cvpr52729.2023.01433. URL http://dx.doi.org/10.1109/cvpr52729.2023.01433.

[60] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-Across-Room: Multilingual Vision-and-Language Navigation with Dense Spatiotemporal Grounding. In *EMNLP*, 2020. doi: 10.18653/v1/2020.emnlp-main.356. URL http://dx.doi.org/10.18653/v1/2020.emnlp-main.356.

[61] Yuxuan Kuang, Hai Lin, and Meng Jiang. OpenFMNav: Towards Open-Set Zero-Shot Object Navigation via Vision-Language Foundation Models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 338–351, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.24. URL https://aclanthology.org/2024.findings-naacl.24.

[62] Lars Kunze, Keerthi Kumar Doreswamy, and Nick Hawes. Using Qualitative Spatial Relations for indirect object search. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2014. doi: 10.1109/icra.2014.6906604. URL http://dx.doi.org/10.1109/icra.2014.6906604.

[63] Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, R. Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten digit recognition with a back-propagation network. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann, 1989. URL https://proceedings.neurips.cc/paper_files/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf.

[64] Jongmin Lee, Geon-Hyeong Kim, Pascal Poupart, and Kee-Eung Kim. Monte-Carlo Tree Search for Constrained POMDPs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. URL https://dl.acm.org/doi/10.5555/3327757.3327889.

[65] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19730–19742. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/li23q.html.

[66] Xiaofan Li, Zhizhong Zhang, Xin Tan, Chengwei Chen, Yanyun Qu, Yuan Xie, and Lizhuang Ma. PromptAD: Learning Prompts with only Normal Samples for Few-Shot Anomaly Detection. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 16848–16858.

IEEE, June 2024. doi: 10.1109/cvpr52733.2024.01594. URL http://dx.doi.org/10.1109/cvpr52733.2024.01594.

[67] Xiwen Liang, Fengda Zhu, Yi Zhu, Bingqian Lin, Bing Wang, and Xiaodan Liang. Contrastive Instruction-Trajectory Learning for Vision-Language Navigation. *AAAI*, 36(2):1592–1600, Jun. 2022. doi: 10.1609/aaai.v36i2.20050. URL https://ojs.aaai.org/index.php/AAAI/article/view/20050.

[68] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual Instruction Tuning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 34892–34916. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/6dcf277ea32ce3288914faf369fe6de0-Paper-Conference.pdf.

[69] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge, January 2024. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

[70] Qianyi Liu, Siqi Zhang, Yanyuan Qiao, Junyou Zhu, Xiang Li, Longteng Guo, Qunbo Wang, Xingjian He, Qi Wu, and Jing Liu. GroundingMate: Aiding Object Grounding for Goal-Oriented Vision-and-Language Navigation. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pages 1775–1784, February 2025. URL https://openaccess.thecvf.com/content/WACV2025/html/Liu_GroundingMate_Aiding_Object_Grounding_for_Goal-Oriented_Vision-and-Language_Navigation_WACV_2025_paper.html.

[71] Shaopeng Liu and Guohui Tian. A High-efficient Training Strategy for Deep Q-learning Network Used in Robot Active Object Detection. In *2022 12th International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*. IEEE, July 2022. doi: 10.1109/cyber55403.2022.9907473. URL http://dx.doi.org/10.1109/cyber55403.2022.9907473.

[72] Shaopeng Liu, Guohui Tian, Yongcheng Cui, and Xuyang Shao. A deep Q-learning network based active object detection model with a novel training algorithm for service robots. *Frontiers of Information Technology & Electronic Engineering*, 23(11):1673–1683, September 2022. ISSN 2095-9230. doi: 10.1631/fitee.2200109. URL http://dx.doi.org/10.1631/fitee.2200109.

[73] Shi Liu, Kecheng Zheng, and Wei Chen. Paying More Attention to Image: A Training-Free Method for Alleviating Hallucination in LVLMs. In *Computer Vision - ECCV 2024*. Springer Nature Switzerland, 2025. URL https://link.springer.com/chapter/10.1007/978-3-031-73010-8_8.

[74] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with Grounded Pre-training for Open-Set

Object Detection. In *Computer Vision – ECCV 2024*, page 38–55. Springer Nature Switzerland, November 2024. ISBN 9783031729706. doi: 10.1007/978-3-031-72970-6_3. URL http://dx.doi.org/10.1007/978-3-031-72970-6_3.

[75] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based Out-of-distribution Detection. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21464–21475. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/f5496252609c43eb8a3d147ab9b9c006-Paper.pdf.

[76] Xiulong Liu, Sudipta Paul, Moitreya Chatterjee, and Anoop Cherian. CAVEN: An Embodied Conversational Agent for Efficient Audio-Visual Navigation in Noisy Environments. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(4):3765–3773, Mar. 2024. doi: 10.1609/aaai.v38i4.28167. URL https://ojs.aaai.org/index.php/AAAI/article/view/28167.

[77] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/c74d97b01eae257e44aa9d5bade97baf-Paper.pdf.

[78] Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. ZSON: Zero-Shot Object-Goal Navigation using Multimodal Goal Embeddings. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 32340–32352. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/d0b8f0c8f79d3a621af945cafb669f4b-Paper-Conference.pdf.

[79] Arjun Majumdar, Fei Xia, Brian Ichter, Dhruv Batra, and Leonidas Guibas. FindThis: Language-Driven Object Disambiguation in Indoor Environments. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 1335–1347. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/majumdar23a.html.

[80] Giulio Mazzi, Alberto Castellini, and Alessandro Farinelli. Identification of Unexpected Decisions in Partially Observable Monte-Carlo Planning: A Rule-Based Approach. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '21, page 889–897, Richland, SC, 2021. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450383073. URL https://dl.acm.org/doi/10.5555/3463952.3464058.

[81] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software*, 3(29):861, September 2018. ISSN 2475-9066. doi: 10.21105/joss.00861. URL http://dx.doi.org/10.21105/joss.00861.

[82] Scott Mayer McKinney, Marcin Sieniek, Varun Godbole, Jonathan Godwin, Natasha Antropova, Hutan Ashrafian, Trevor Back, Mary Chesus, Greg S. Corrado, Ara Darzi, Mozziyar Etemadi, Florencia Garcia-Vicente, Fiona J. Gilbert, Mark Halling-Brown, Demis Hassabis, Sunny Jansen, Alan Karthikesalingam, Christopher J. Kelly, Dominic King, Joseph R. Ledsam, David Melnick, Hormuz Mostofi, Lily Peng, Joshua Jay Reicher, Bernardino Romera-Paredes, Richard Sidebottom, Mustafa Suleyman, Daniel Tse, Kenneth C. Young, Jeffrey De Fauw, and Shravya Shetty. International evaluation of an ai system for breast cancer screening. *Nature*, 577(7788):89–94, Jan 2020. ISSN 1476-4687. doi: 10.1038/s41586-019-1799-6. URL https://doi.org/10.1038/s41586-019-1799-6.

[83] Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andy Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, et al. Learning to Navigate in Complex Environments. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://openreview.net/forum?id=SJMGPrcle.

[84] Morgan Stanley Research. Humanoids: Investment Implications of Embodied AI, June 2024. URL https://www.futuremanagementgroup.com/wp-content/uploads/240626-Humanoid-Robots-Morgan-Stanley.pdf. Morgan Stanley Global Foundation Report. Accessed July 15, 2025.

[85] Lia Morra, Alberto Azzari, Letizia Bergamasco, Marco Braga, Luigi Capogrosso, Federico Delrio, Giuseppe Di Giacomo, Simone Eiraudo, Giorgia Ghione, Rocco Giudice, Alkis Koudounas, Luca Piano, Daniele Rege Cambrin, Matteo Risso, Marco Rondina, Alessandro Sebastien Russo, Marco Russo, Francesco Taioli, Lorenzo Vaiani, and Chiara Vercellino. Designing Logic Tensor Networks for Visual Sudoku puzzle classification. In *CEUR Workshop Proceedings*, volume 3432, pages 223–232. CEUR, 2023. URL https://hdl.handle.net/11583/2978475.

[86] Arsalan Mousavian, Alexander Toshev, Marek Fiser, Jana Kosecka, Ayzaan Wahid, and James Davidson. Visual representations for semantic target driven navigation. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, May 2019. doi: 10.1109/icra.2019.8793493. URL http://dx.doi.org/10.1109/icra.2019.8793493.

[87] Khanh Nguyen and Hal Daumé III. "Help, Anna! Visual Navigation with Natural Multimodal Assistance via Retrospective Curiosity-Encouraging Imitation Learning". In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 684–695, Hong

Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1063. URL https://aclanthology.org/D19-1063.

[88] Khanh Nguyen and Hal Daumé III. Help, Anna! Visual Navigation with Natural Multimodal Assistance via Retrospective Curiosity-Encouraging Imitation Learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 2019. doi: 10.18653/v1/d19-1063. URL http://dx.doi.org/10.18653/v1/d19-1063.

[89] Khanh Nguyen, Debadeepta Dey, and Bill Brockett, Chrnd Dolan. Vision-Based Navigation With Language-Based Assistance via Imitation Learning With Indirect Intervention. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2019. doi: 10.1109/cvpr.2019.01281. URL http://dx.doi.org/10.1109/cvpr.2019.01281.

[90] Khanh X Nguyen, Yonatan Bisk, and Hal Daumé Iii. A Framework for Learning to Request Rich and Contextually Useful Information from Humans. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 16553–16568. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/nguyen22a.html.

[91] Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. LLMs Know More Than They Show: On the Intrinsic Representation of LLM Hallucinations. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=KRnsX5Em3W.

[92] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

[93] Jeongeun Park, Seungwon Lim, Joonhyung Lee, Sangbeom Park, Minsuk Chang, Youngjae Yu, and Sungjoon Choi. CLARA: Classifying and Disambiguating User Commands for Reliable Interactive Robotic Agents. *IEEE Robotics and Automation Letters*, 9(2):1059–1066, 2024. URL 10.1109/LRA.2023.3338514.

[94] Sudipta Paul, Amit Roy-Chowdhury, and Anoop Cherian. AVLEN: Audio-Visual-Language Embodied Navigation in 3D Environments. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 6236–6249. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/28f699175783a2c828ae74d53dd3da20-Paper-Conference.pdf.

[95] Yusu Qian, Haotian Zhang, Yinfei Yang, and Zhe Gan. How Easy is It to Fool Your Multimodal LLMs? An Empirical Analysis on Deceptive Prompts. In *Neurips Safe Generative AI Workshop 2024*, 2024. URL https://openreview.net/forum?id=BGY6LWN8bh.

[96] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

[97] 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 *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=-v4OuqNs5P.

[98] Santhosh Kumar Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. PONI: Potential Functions for ObjectGoal Navigation with Interaction-free Learning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022. doi: 10.1109/cvpr52688.2022.01832. URL http://dx.doi.org/10.1109/cvpr52688.2022.01832.

[99] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical Text-Conditional Image Generation with CLIP Latents, 2022. URL https://arxiv.org/abs/2204.06125.

[100] Ram Ramrakhya, Dhruv Batra, Erik Wijmans, and Abhishek Das. PIRLNav: Pretraining with Imitation and RL Finetuning for OBJECTNAV. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023. doi: 10.1109/cvpr52729.2023.01716. URL http://dx.doi.org/10.1109/cvpr52729.2023.01716.

[101] Niyati Rawal, Roberto Bigazzi, Lorenzo Baraldi, and Rita Cucchiara. UNMuTe: Unifying Navigation and Multimodal Dialogue-like Text Generation.

arXiv preprint arXiv:2408.04423, 2024. URL https://arxiv.org/abs/2408.04423.

[102] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

[103] Allen Z. Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, Zhenjia Xu, Dorsa Sadigh, Andy Zeng, and Anirudha Majumdar. Robots That Ask For Help: Uncertainty Alignment for Large Language Model Planners. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=4ZK8ODNyFXx.

[104] D. A. Sasi Kiran, Kritika Anand, Chaitanya Kharyal, Gulshan Kumar, Nandiraju Gireesh, Snehasis Banerjee, Ruddra Dev Roychoudhury, Mohan Sridharan, Brojeshwar Bhowmick, and Madhava Krishna. Spatial Relation Graph and Graph Convolutional Network for Object Goal Navigation. In *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*. IEEE, August 2022. doi: 10.1109/case49997.2022.9926534. URL http://dx.doi.org/10.1109/case49997.2022.9926534.

[105] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, page 9338–9346. IEEE, October 2019. doi: 10.1109/iccv.2019.00943. URL http://dx.doi.org/10.1109/iccv.2019.00943.

[106] Jan Fabian Schmid, Mikko Lauri, and Simone Frintrop. Explore, Approach, and Terminate: Evaluating Subtasks in Active Visual Object Search Based on Deep Reinforcement Learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, November 2019. doi: 10.1109/iros40897.2019.8967805. URL http://dx.doi.org/10.1109/iros40897.2019.8967805.

[107] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[108] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models, 2024. URL https://arxiv.org/abs/2402.03300.

[109] Ying Shen, Daniel Bis, Cynthia Lu, and Ismini Lourentzou. ELBA: Learning by Asking for Embodied Visual Navigation and Task Completion. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pages 5177–5186, February 2025. URL https://openaccess.thecvf.com/content/WACV2025/papers/Shen_ELBA_Learning_by_Asking_for_Embodied_Visual_Navigation_and_Task_WACV_2025_paper.pdf.

[110] David Silver and Joel Veness. Monte-Carlo Planning in Large POMDPs. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. URL https://proceedings.neurips.cc/paper_files/paper/2010/file/edfbe1afcf9246bb0d40eb4d8027d90f-Paper.pdf.

[111] Kunal Pratap Singh, Luca Weihs, Alvaro Herrasti, Jonghyun Choi, Aniruddha Kembhavi, and Roozbeh Mottaghi. Ask4Help: Learning to Leverage an Expert for Embodied Tasks. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 16221–16232. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/674ad201bc8fa74b3c9979230aa0c63b-Paper-Conference.pdf.

[112] Xander Sun, Louis Lau, Hoyard Zhi, Ronghe Qiu, and Junwei Liang. Prioritized Semantic Learning for Zero-shot Instance Navigation. In *Computer Vision - ECCV 2024*. Springer Nature Switzerland, 2025. URL https://link.springer.com/chapter/10.1007/978-3-031-73254-6_10.

[113] Petr Svestka and Mark H. Overmars. Motion Planning for Carlike Robots Using a Probabilistic Learning Approach. *Int. J. Robotics Res.*, 16(2):119–143, 1997. doi: 10.1177/027836499701600201. URL https://doi.org/10.1177/027836499701600201.

[114] Andrew Szot, Bogdan Mazoure, Omar Attia, Aleksei Timofeev, Harsh Agrawal, Devon Hjelm, Zhe Gan, Zsolt Kira, and Alexander Toshev. From Multimodal LLMs to Generalist Embodied Agents: Methods and Lessons. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10644–10655, June 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Szot_From_Multimodal_LLMs_to_Generalist_Embodied_Agents_Methods_and_Lessons_CVPR_2025_paper.html.

[115] Francesco Taioli, Federico Cunico, Federico Girella, Riccardo Bologna, Alessandro Farinelli, and Marco Cristani. Language-Enhanced RNR-Map: Querying Renderable Neural Radiance Field maps with natural language. In *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 4671–4676, Los Alamitos, CA, USA, October 2023. IEEE Computer Society. URL https://doi.ieeecomputersociety.org/10.1109/ICCVW60793.2023.00504.

[116] Francesco Taioli, Francesco Giuliari, Yiming Wang, Riccardo Berra, Alberto Castellini, Alessio Del Bue, Alessandro Farinelli, Marco Cristani, and Francesco Setti. Unsupervised Active Visual Search With Monte Carlo Planning Under Uncertain Detections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):11047–11058, 2024. URL https://doi.org/10.1109/TPAMI.2024.3451994.

[117] Francesco Taioli, Stefano Rosa, Alberto Castellini, Lorenzo Natale, Alessio Del Bue, Alessandro Farinelli, Marco Cristani, and Yiming Wang. I2EDL: Interactive Instruction Error Detection and Localization. In *2024 33rd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1872–1877, 2024. URL https://doi.org/10.1109/RO-MAN60168.2024.10731349.

[118] Francesco Taioli, Stefano Rosa, Alberto Castellini, Lorenzo Natale, Alessio Del Bue, Alessandro Farinelli, Marco Cristani, and Yiming Wang. Mind the Error! Detection and Localization of Instruction Errors in Vision-and-Language Navigation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12993–13000, 2024. URL https://doi.org/10.1109/IROS58592.2024.10801822.

[119] 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 (ICCV)*, pages 18781–18792, October 2025. URL https://openaccess.thecvf.com/content/ICCV2025/papers/Taioli_Collaborative_Instance_Object_Navigation_Leveraging_Uncertainty-Awareness_to_Minimize_Human-Agent_Dialogues_ICCV_2025_paper.pdf.

[120] Hao Tan and Mohit Bansal. "LXMERT: Learning Cross-Modality Encoder Representations from Transformers". In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1514. URL https://aclanthology.org/D19-1514/.

[121] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to Navigate Unseen Environments: Back Translation with Environmental Dropout. In *NAACL*, pages 2610–2621, Minneapolis, Minnesota, June 2019. doi: 10.18653/v1/N19-1268. URL https://aclanthology.org/N19-1268.

[122] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, and et al. Gemini: A Family of Highly Capable Multimodal Models, 2025. URL https://arxiv.org/abs/2312.11805.

[123] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-Dialog Navigation. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 394–406. PMLR, 30 Oct–01 Nov 2020. URL https://proceedings.mlr.press/v100/thomason20a.html.

[124] Sebastian Thrun. Monte Carlo pomdps. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'99, page 1064–1070, Cambridge, MA, USA, 1999. MIT Press. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/299570476c6f0309545110c592b6a63b-Paper.pdf.

[125] Ekin Tiu, Ellie Talius, Pujan Patel, Curtis P. Langlotz, Andrew Y. Ng, and Pranav Rajpurkar. Expert-level detection of pathologies from unannotated chest x-ray images via self-supervised learning. *Nature Biomedical Engineering*, 6(12):1399–1406, Dec 2022. ISSN 2157-846X. doi: 10.1038/s41551-022-00936-9. URL https://doi.org/10.1038/s41551-022-00936-9.

[126] Andrea Toaiari, Federico Cunico, Francesco Taioli, Ariel Caputo, Gloria Menegaz, Andrea Giachetti, Giovanni Maria Farinella, and Marco Cristani. "SCENE-pathy: Capturing the Visual Selective Attention of People Towards Scene Elements". In Gian Luca Foresti, Andrea Fusiello, and Edwin Hancock, editors, *Image Analysis and Processing – ICIAP 2023*, pages 352–363, Cham, 2023. Springer Nature Switzerland. ISBN 978-3-031-43148-7. URL https://doi.org/10.1007/978-3-031-43148-7_30.

[127] Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes Wide Shut? Exploring the Visual Shortcomings of Multimodal LLMs. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 9568–9578. IEEE, June 2024. doi: 10.1109/cvpr52733.2024.00914. URL http://dx.doi.org/10.1109/cvpr52733.2024.00914.

[128] Tao Tu, Shekoofeh Azizi, Danny Driess, Mike Schaekermann, Mohamed Amin, Pi-Chuan Chang, Andrew Carroll, Charles Lau, Ryutaro Tanno, Ira Ktena, Anil Palepu, Basil Mustafa, Aakanksha Chowdhery, Yun Liu, Simon Kornblith, David Fleet, Philip Mansfield, Sushant Prakash, Renee Wong, Sunny Virmani, Christopher Semturs, S. Sara Mahdavi, Bradley Green, Ewa Dominowska, Blaise Aguera y Arcas, Joelle Barral, Dale Webster, Greg S. Corrado, Yossi Matias, Karan Singhal, Pete Florence, Alan Karthikesalingam, and Vivek Natarajan. Towards generalist biomedical ai. *NEJM AI*, 1(3): AIoa2300138, 2024. doi: 10.1056/AIoa2300138. URL https://ai.nejm.org/doi/full/10.1056/AIoa2300138.

[129] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates,

Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[130] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 3156–3164. IEEE, June 2015. doi: 10.1109/cvpr.2015.7298935. URL http://dx.doi.org/10.1109/cvpr.2015.7298935.

[131] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced Cross-Modal Matching and Self-Supervised Imitation Learning for Vision-Language Navigation. In *CVPR*, June 2019. doi: 10.1109/cvpr.2019.00679. URL http://dx.doi.org/10.1109/cvpr.2019.00679.

[132] Yiming Wang, Francesco Giuliari, Riccardo Berra, Alberto Castellini, Alessio Del Bue, Alessandro Farinelli, Marco Cristani, and Francesco Setti. POMP: Pomcp-based Online Motion Planning for active visual search in indoor environments. In *Proc. of British Machine Vision Conference (BMVC)*, 2020. doi: https://doi.org/10.48550/arXiv.2009.08140. URL https://doi.org/10.48550/arXiv.2009.08140.

[133] Z. Wang, X. Li, J. Yang, Y. Liu, and S. Jiang. GridMM: Grid Memory Map for Vision-and-Language Navigation. In *ICCV*, pages 15579–15590, oct 2023. doi: 10.1109/ICCV51070.2023.01432. URL https://doi.ieeecomputersociety.org/10.1109/ICCV51070.2023.01432.

[134] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

[135] Spencer Whitehead, Suzanne Petryk, Vedaad Shakib, Joseph Gonzalez, Trevor Darrell, Anna Rohrbach, and Marcus Rohrbach. Reliable Visual Question Answering: Abstain Rather Than Answer Incorrectly. In *Computer Vision – ECCV 2022*, page 148–166. Springer Nature Switzerland, 2022. ISBN 9783031200595. doi: 10.1007/978-3-031-20059-5_9. URL http://dx.doi.org/10.1007/978-3-031-20059-5_9.

[136] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. DD-PPO: learning near-perfect pointgoal navigators from 2.5 billion frames. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=H1gX8C4YPr.

[137] Yichen Xie, Runsheng Xu, Tong He, Jyh-Jing Hwang, Katie Luo, Jingwei Ji, Hubert Lin, Letian Chen, Yiren Lu, Zhaoqi Leng, Dragomir Anguelov, and Mingxing Tan. S4-driver: Scalable self-supervised driving multimodal large language model with spatio-temporal visual representation. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 1622–1632, June 2025. URL https://doi.org/10.48550/arXiv.2505.24139.

[138] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, pages 146–151, 1997. URL 10.1109/CIRA.1997.613851.

[139] Zijiao Yang, Arjun Majumdar, and Stefan Lee. Behavioral Analysis of Vision-and-Language Navigation Agents. In *CVPR*, June 2023. doi: 10.1109/cvpr52729.2023.00253. URL http://dx.doi.org/10.1109/cvpr52729.2023.00253.

[140] Xin Ye, Zhe Lin, Joon-Young Lee, Jianming Zhang, Shibin Zheng, and Yezhou Yang. GAPLE: Generalizable Approaching Policy LEarning for Robotic Object Searching in Indoor Environment. *IEEE Robotics and Automation Letters*, 4(4):4003–4010, October 2019. ISSN 2377-3774. doi: 10.1109/lra.2019.2930426. URL http://dx.doi.org/10.1109/lra.2019.2930426.

[141] 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)*, volume 35, page 42–48. IEEE, May 2024. doi: 10.1109/icra57147.2024.10610712. URL http://dx.doi.org/10.1109/icra57147.2024.10610712.

[142] Naoki Yokoyama, Ram Ramrakhya, Abhishek Das, Dhruv Batra, and Sehoon Ha. HM3D-OVON: A Dataset and Benchmark for Open-Vocabulary Object Goal Navigation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, page 5543–5550. IEEE, October 2024. doi: 10.1109/iros58592.2024.10802709. URL http://dx.doi.org/10.1109/iros58592.2024.10802709.

[143] 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)*. IEEE, October 2023. doi: 10.1109/iros55552.2023.10342512. URL http://dx.doi.org/10.1109/iros55552.2023.10342512.

[144] 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, 2024. doi: 10.1109/IROS58592.2024.10802670. URL 10.1109/IROS58592.2024.10802670.

[145] Michael JQ Zhang and Eunsol Choi. "Clarify When Necessary: Resolving Ambiguity Through Interaction with LMs". In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5526–5543, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.306. URL https://aclanthology.org/2025.findings-naacl.306/.

[146] Yubo Zhang, Hao Tan, and Mohit Bansal. Diagnosing the Environment Bias in Vision-and-Language Navigation. In *IJCAI*, IJCAI-PRICAI-2020, July 2020. doi: 10.24963/ijcai.2020/124. URL http://dx.doi.org/10.24963/ijcai.2020/124.

[147] Ming Zhao, Peter Anderson, Vihan Jain, Su Wang, Alexander Ku, Jason Baldridge, and Eugene Ie. On the Evaluation of Vision-and-Language Navigation Instructions. In *EACL*, pages 1302–1316, Online, April 2021. doi: 10.18653/v1/2021.eacl-main.111. URL https://aclanthology.org/2021.eacl-main.111.

[148] Qinyu Zhao, Ming Xu, Kartik Gupta, Akshay Asthana, Liang Zheng, and Stephen Gould. The First to Know: How Token Distributions Reveal Hidden Knowledge in Large Vision-Language Models? In *Computer Vision - ECCV 2024*. Springer Nature Switzerland, 2025. URL https://link.springer.com/chapter/10.1007/978-3-031-73195-2_8.

[149] 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 Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 42829–42842. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/zhou23r.html.

[150] Deyao Zhu, Jun Chen, Kilichbek Haydarov, Xiaoqian Shen, Wenxuan Zhang, and Mohamed Elhoseiny. ChatGPT Asks, BLIP-2 Answers: Automatic Questioning Towards Enriched Visual Descriptions. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=1LoVwFkZNo.

[151] Wanrong Zhu, Yuankai Qi, Pradyumna Narayana, Kazoo Sone, Sugato Basu, Xin Wang, Qi Wu, Miguel Eckstein, and William Yang Wang. Diagnosing Vision-and-Language Navigation: What Really Matters. In *NAACL*, pages 5981–5993, Seattle, United States, July 2022. doi: 10.18653/v1/2022.naacl-main.438. URL https://aclanthology.org/2022.naacl-main.438.

[152] Yi Zhu, Yue Weng, Fengda Zhu, Xiaodan Liang, Qixiang Ye, Yutong Lu, and Jianbin Jiao. Self-Motivated Communication Agent for Real-World Vision-Dialog Navigation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, October 2021. doi: 10.1109/iccv48922.2021.00162. URL http://dx.doi.org/10.1109/iccv48922.2021.00162.

Appendix A

Collaborative Instance Navigation

A.1 AIUTA: Prompts

A.1.1 P_{init} - Initial Description

```
P_init = """Describe the {target_object} in the provided image.
    """
```

A.1.2 $P_{details}$ - Gather Additional Information

```
P_details = """You are an intelligent embodied agent equipped
    with an RGB sensor, an object detector, and a Visual
    Question Answering (VQA) model.
Your task is to explore an indoor environment to find a
    specific target {target_object}.
The detector has identified a {target_object}. The VQA model
    has provided the following description of the scene:

<START_OF_DESCRIPTION>
{distractor_object_description}
<END_OF_DESCRIPTION>

Based on your past interactions with the user, you know the
    following facts about the target picture:
<START_TARGET_PICTURE_FACTS>
{facts_about_the_target_picture}
<END_TARGET_PICTURE_FACTS>
```

```

Your task is to:
- ask more question to the VQA model on the detected {
    target_object} to maximize information gain.

Ensure your output follows the following format:

YAML_START # must be present to get the information back
attributes_of_the_image:
     <attribute name>: "<attribute value>" # summarize all the
    known attributes from the description, enclosed in " "
questions:
        <question_number>: "<question content>"
YAML_END # must be present to get the information back

Provide your reasoning step-by-step, after the YAML_END tag."""
```

A.1.3 P_{check} - Check detection with LVML

```
 P_check = """Does the image contain a {target_object}? Answer
    with Yes, No or ?=I don't know."""
```

A.1.4 $P_{self question}$ - Extract attributes and generate Self-Questions

```
P_ATTRIBUTES_AND_SELF_QUESTIONS = """
You are an intelligent embodied agent equipped with an RGB
    sensor, an object detector, and a Visual Question Answering
    (VQA) model. Your task is to explore an indoor environment
    to find a specific target {target_object}.
The detector has identified a {target_object}. The VQA model
    has provided the following description of the scene:

<START_OF_DESCRIPTION>
{distractor_object_description}
<END_OF_DESCRIPTION>

Based on your past interactions with the user, you know the
    following facts about the target picture: <
    START_TARGET_PICTURE_FACTS> {facts_about_the_target_picture}
     <END_TARGET_PICTURE_FACTS>

```

```
Assume that the detected image description contains hallucinations. Your goal is to verify every attribute of the detected {target_object} description through questions. Formally:
- Detect possible hallucinations in the VQA model's description
- Get more information about the detected object.
Every question should be in this format: "<question content>? You must answer only with Yes, No, or ?=I don't know." This allows us to assess the likelihood of the answers.


Ensure your output follows the following format:
YAML_START # must be present to get the information back
attributes_of_the_image:
    <attribute name>: "<attribute value>" # summarize all the known attributes from the description, enclosed in " "

questions_for_detected_object: # question for the detected object, if any
    <Question number>:  "<question>? You must answer only with Yes, No, or ?=I don't know."
reasoning_for_detected_object:
    <Question number>: <reasoning>
YAML_END # must be present to get the information back

Provide your reasoning step-by-step, after the YAML_END tag."""
```

A.1.5 $P_{refined}$ - Refined image description

```
P_refined = """
You are an intelligent embodied agent equipped with an RGB sensor, an object detector, and a Visual Question Answering (VQA) model.
Your task is to refine an image description based on certainty estimates and user interactions.

Scenario:
The detector has identified a scene with a {target_object}. The VQA model provided this initial scene description:

<START_OF_DESCRIPTION>
{distractor_object_description}
```

```
<END_OF_DESCRIPTION>


Questions asked and responses:
<START_QUESTION_AND_RESPONSES>
{list_questions_answers_uncertainty_labels}
<END_QUESTION_AND_RESPONSES>

Task:
Using the questions/answer pairs with uncertainty labels,
   refine the image description.
Since we have to find a {target_object}, put enphasis on it. Do
    not include in the description information that is labeled
   as uncertain.

Ensure your response follows the format below:
YAML_START # must be present to get the information back
attributes_of_the_image:
    <attribute name>: "<attribute value>" # summarize all the
   known attributes from the description, enclosed in " "
image_description_refined: <insert refined description>  #
   Ensure that the string does not contain a newline (\n) after
    the tag image_description_refined:
YAML_END # must be present to get the information back

Provide your reasoning step-by-step, after the YAML_END tag."""
```

A.1.6 P_{score} - Alignment score

```
P_score = """
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 {target_object} based on a
   scene description and prior knowledge of the target.

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

<START_OF_DESCRIPTION>
{distractor_object_description}
<END_OF_DESCRIPTION>
```

```
11
12 Target object information:
13 Based on previous interactions, you know the target picture has
       the following characteristics:
14 <START_TARGET_PICTURE_FACTS>
15 {facts_about_the_target_picture}
16 <END_TARGET_PICTURE_FACTS>
17
18 Task:
19 1. Similarity analysis.
20 Analyze how closely the detected scene description aligns with
     the known facts about the target {target_object}. Provide a
     similarity score between 0 and 10, where:
21 - 0 = The detected {target_object} is not the target object.
22 - 10 = The detected {target_object} is definitely the target
     object.
23 - If no information about the target is available, the score
     should be -1.
24
25 2. Question Generation:
26 - The question is for the target object, not the detected one.
27 - Ask exactly one specific, relevant, and human-answerable
     question related to the target object that maximizes
     information gain for identifying the target {target_object}.
28 - Do not ask speculative or irrelevant questions
29 - 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.
30
31 Ensure your response follows the format below:
32 YAML_START # must be present to get the information back
33 similarity_score: <similarity score>
34 questions:
35     <question_number>: <question_content>
36 YAML_END # must be present to get the information back
37
38 Provide your reasoning step-by-step for the similarity score
     and questions, after the YAML_END tag."""
```

A.2 AIUTA: Algorithm

Algorithm 1 outlines the complete AIUTA pipeline. Upon detecting a candidate object, AIUTA first invokes the *Self Questioner* (Alg. A.2.1) module to obtain an accurate and detailed understanding of the observed object and to reduce inaccuracies and hallucinations, obtaining a refined observation description $S_{refined}$. Then, with the known facts about the target instance and the refined description, AIUTA invokes the *Interaction Trigger* module (Alg. A.2.2) for up to 4 iterations rounds. Within each interaction round, if AIUTA returns the `STOP` action, then the policy π terminates the navigation since the target instance is found; otherwise, the policy π continues the navigation process.

Algorithm 1 AIUTA

Require: Target object facts F, Observation O_t, policy π, Candidate Object Detection, Max Iteration number

▷ *Upon candidate object detection*

$S_{refined} \leftarrow$ Self_Questioner(F, O_t) ▷ *enrich details and reduce inaccuracy, obtain a refined description*

if $S_{refined} =$ " " **then**

  π(CONTINUE_EXPLORING) ▷ *VQA detection failed, Signal to policy π to continue exploration*

for each iteration in Max_Iteration_Number **do**

  aiuta_action $\leftarrow$ Interaction_Trigger$(F, S_{refined})$

  if aiuta_action = STOP **then**

    π(STOP) ▷ *Signal to policy π that the object is found! Terminate exploration*

  else

    π(CONTINUE_EXPLORING) ▷ *Signal to policy π to continue exploration*

A.2.1 Self Questioner

Algorithm 2 Self Questioner Module

Require: Target object facts F, Uncertainty Threshold τ, Observation O_t, $P_{init}, P_{details}, P_{check}, P_{self questions}, P_{refined}$

Step 1: Detailed Detection Description, from S_{init} to S_{enriched}
Initial scene description: $S_{\text{init}} \leftarrow \text{VLM}(O_t, P_{\text{init}})$
Self-generate questions to enrich description
$Q_{a\to a}^{details} \leftarrow \text{LLM}(P_{\text{details}}, S_{\text{init}}, F)$
for each question q_j in $Q_{a\to a}^{details}$ **do**
    $r_{a\to a} \leftarrow \text{VLM}(O_t, q_j)$ ▷ *Get answers*
    $S_{\text{init}} \leftarrow \text{concatenate}(S_{\text{init}}, r_{a\to a})$
$S_{\text{enriched}} \leftarrow S_{\text{init}}$ ▷ *Updated scene description*
Step 2: Perception Uncertainty Estimation
$(r_{\text{check}}, u_{\text{check}}) \leftarrow \text{VLM}(O_t, P_{\text{check}})$ ▷ *Check detection with uncertainty*
if NOT (r_{check} = "Yes" AND u_{check} = "Certain") **then**
    return " " ▷ *empty string, thus continue exploring*
$Q_{a\to a}^{attribute} \leftarrow \text{LLM}(P_{\text{self questions}}, F, S_{\text{enriched}})$ ▷ *Generate self-questions to verify attributes*
Container $\leftarrow \{\}$ ▷ *Store question, answer, uncertainty*
for each question q_j in $Q_{a\to a}^{attribute}$ **do**
    $(r_j, u_j) \leftarrow \text{VLM}(O_t, q_j)$ ▷ *Get answers and uncertainties*
    Container $\leftarrow \text{concatenate}(\text{Container}, \{q_j, r_j, u_j\})$
Step 3: Detection Description Refinement
$S_{\text{refined}} \leftarrow \text{LLM}(P_{\text{refined}}, \text{Container}, S_{\text{enriched}})$ ▷ *Filter out uncertain attributes*
return S_{refined}

A.2.2 Interaction Trigger

Algorithm 3 Interaction Trigger

Require: Target object facts F, Refined observation description S_{refined}, P_{score}, τ_{stop} and τ_{skip}

$(s, q_{a\to u}) \leftarrow \text{LLM}(P_{score}, S_{\text{refined}}, F)$ ▷ *get alignment score s, and question for the human* $q_{a\to u}$
if $s \geq \tau_{stop}$ **then**
    return STOP ▷ *target found, stop navigation.*
else if $s < \tau_{skip}$ **then**
    return CONTINUE_EXPLORING ▷ *skip the question and continue exploring*
else
    $r_{u\to a} \leftarrow \text{Ask\_Human}(q_{a\to u})$ ▷ *posing clarifying question* $q_{a\to u}$ *from the agent to the human.*
    $F \leftarrow \text{Update\_Facts}(F, r_{u\to a})$ ▷ *update target object facts F*