FH University of Applied Sciences TECHNIKUM WIEN

MASTER THESIS

Thesis submitted in fulfillment of the requirements for the degree of Master of Science in Engineering at the University of Applied Sciences Technikum Wien - Degree Program Mechatronics/Robotics

SAGE: Multi object semantic aware guided exploration with persistent memory

By: Kevin Eppacher, BSc

Student Number: 2310331013

Supervisors: Simon Schwaiger, MSc
FH-Prof. Dr.techn. Mohamed Aburaia MSc.

Vienna, January 31, 2026

This work was conducted in the context of the project “Stadt Wien Kompetenzteam für Drohnentechnik in der Fachhochschulausbildung” (project number MA23 35-02, financed by the Department MA23 for Economic Affairs, Labour and Statistics of the City of Vienna).

Funded by City of Vienna | Economic Affairs, Labour and Statistics

Drohn FH

Declaration

“As author and creator of this work to hand, I confirm with my signature knowledge of the relevant copyright regulations governed by higher education acts (see Urheberrechtsgesetz /Austrian copyright law as amended as well as the Statute on Studies Act Provisions / Examination Regulations of the UAS Technikum Wien as amended).

I hereby declare that I completed the present work independently and that any ideas, whether written by others or by myself, have been fully sourced and referenced. I am aware of any consequences I may face on the part of the degree program director if there should be evidence of missing autonomy and independence or evidence of any intent to fraudulently achieve a pass mark for this work (see Statute on Studies Act Provisions / Examination Regulations of the UAS Technikum Wien as amended).

I further declare that up to this date I have not published the work to hand nor have I presented it to another examination board in the same or similar form. I affirm that the version submitted matches the version in the upload tool.“

Vienna, January 31, 2026 Signature

Kurzfassung

Open-Vocabulary Semantic Exploration erfordert, dass mobile Roboter nutzerspezifische Zielobjekte in zuvor unbekannten Umgebungen finden, während sie mit partieller Beobachtbarkeit, perzeptueller Mehrdeutigkeit und begrenzten Onboard-Rechenressourcen umgehen. Bestehende Ansätze basieren entweder auf Reinforcement Learning mit umfangreichem Simulationstraining oder nutzen Zero-Shot Vision-Language-Modelle ohne persistentes Gedächtnis, was zu ineffizienter Exploration und fragilen Entscheidungen bei verrauschten Beobachtungen führen kann. Diese Arbeit präsentiert SAGE (Semantic-Aware Guided Exploration), ein hybrides Framework für semantische Exploration, das (i) frontier-basierte semantische Exploration auf Basis von Vision-Language-Ähnlichkeit, (ii) ein persistentes dreidimensionales semantisches Gedächtnis auf Grundlage von OpenFusion zur langfristigen Exploitation sowie (iii) eine Fusion mehrerer semantischer Quellen zur robusten Zielbestätigung kombiniert. Die Methode integriert geometrische Machbarkeit und Navigationsrestriktionen in die Frontier-Bewertung und fusioniert Detektor-Konfidenz, Vision-Language-Ähnlichkeit und Evidenz aus dem semantischen Gedächtnis mittels einer Noisy-Or-Formulierung, um insbesondere bei konservativen Entscheidungsschwellen verpasste Detektionen zu reduzieren.

SAGE wird in einem photorealistischen IsaacSim-Setup auf Szenen des Datensatzes Habitat-Matterport 3D Version 2 unter Verwendung eines Mehrziel-Suchprotokolls evaluiert. Die Experimente quantifizieren (1) die End-to-End-Navigationsleistung gegenüber repräsentativen Baselines, (2) die Sensitivität gegenüber der Gewichtung zwischen Exploration und Gedächtnisnutzung, (3) die Robustheit gegenüber der Granularität der semantischen Karte über die OpenFusion Top-k-Retrieval-Tiefe sowie (4) den Effekt der Fusion mehrerer semantischer Quellen auf die Präzisions-Recall-Eigenschaften der Detektion. Die Ergebnisse zeigen innerhalb dieses eigens definierten Evaluationssettings, dass SAGE die höchste Erfolgsgewichtung nach Pfadlänge unter den verglichenen Methoden erreicht, bei gleichzeitig konkurrenzfähiger Erfolgsrate. Weitere Analysen zeigen, dass mittlere Gewichtungen zwischen Exploration und Gedächtnisnutzung den stabilsten Kompromiss zwischen Reaktivität und Exploitation liefern und dass balancierte Exploration die Auswirkungen verrauschter oder übermäßig dichter semantischer Karten abmildert. Die Fusion mehrerer semantischer Quellen reduziert falsch-negative Detektionen an konservativen Betriebspunkten deutlich bei gleichzeitig erhaltener Präzision und unterstützt damit eine zweistufige Verifikationsstrategie im Behavior Tree. Abschließend berichtet eine deploymentsorientierte Analyse den Grafikprozessor-Speicherverbrauch sowie Mehrtakt-Laufzeitcharakteristika.

Schlagworte: Zero-Shot-Objektzielnavigation, Vision-Language-Modelle, frontier-basierte Exploration, 3D-semantische Kartierung, Multi-Source-semantische Fusion

Abstract

Open-vocabulary semantic exploration requires mobile robots to search for open-language targets in previously unseen environments, while coping with partial observability, perceptual ambiguity, and limited onboard compute. Existing approaches either rely on reinforcement learning with extensive training in simulation, or use zero-shot vision-language models without persistent memory, which can lead to inefficient exploration and brittle decisions under noisy observations. This thesis presents SAGE (Semantic-Aware Guided Exploration), a hybrid semantic exploration framework that combines (i) frontier-based semantic exploration driven by vision-language similarity, (ii) persistent three-dimensional semantic memory based on OpenFusion for long-horizon exploitation, and (iii) multi-source semantic fusion for robust target confirmation. The method integrates geometric feasibility and navigational constraints into frontier scoring and fuses detector confidence, vision-language similarity, and memory evidence using a Noisy-Or formulation to reduce missed detections under conservative decision thresholds.

SAGE is evaluated in a photorealistic IsaacSim setup on Habitat-Matterport 3D Version 2 scenes using a multi-object search protocol. Experiments quantify (1) end-to-end navigation performance against representative baselines, (2) sensitivity to the exploration-memory weighting, (3) robustness to semantic map granularity via the OpenFusion top-k retrieval depth, and (4) the effect of multi-source fusion on detection precision-recall characteristics. Results show that, within this custom evaluation setting, SAGE achieves the highest Success weighted by Path Length among the compared methods while maintaining a competitive Success Rate. Further analyses show that intermediate exploration-memory weightings yield the most stable trade-off between reactivity and exploitation, and that balanced exploration mitigates the impact of noisy or overly dense semantic maps. Multi-source fusion substantially reduces false negatives at conservative operating points while maintaining precision, supporting a two-stage verification strategy in the behavior tree. Finally, a deployment-oriented analysis reports GPU memory consumption and multi-rate runtime characteristics.

Keywords: zero-shot object-goal navigation, vision-language models, frontier-based exploration, 3D semantic mapping, multi-source semantic fusion

Acknowledgements

I would like to express my sincere gratitude to Simon Schwaiger, MSc, for his continuous guidance, expertise, and the time he dedicated to supervising this thesis. His constructive feedback, clear perspective, and support throughout the research and writing process were invaluable and greatly contributed to the quality of this work.

I am also thankful to UAS Technikum Vienna for the opportunity to pursue this topic and for providing the necessary resources and infrastructure to carry out the project. In particular, I appreciate the access to the mobile robot platform and supporting equipment that enabled the real-world experiments and made a practical validation of the proposed approach possible.

Thanks go to my fellow students and colleagues for the many discussions, critical questions, and helpful suggestions along the way. Their advice and encouragement, as well as the shared problem-solving during challenging phases, provided both technical insight and much-needed motivation.

I am deeply grateful to my girlfriend, friends, and family for their unwavering moral support, patience, and understanding throughout this demanding period. Their encouragement helped me stay focused and confident, especially during stressful milestones.

Finally, I would like to thank everyone who believed in me and supported me in ways both big and small. This thesis would not have been possible without that trust and encouragement.

Contents

1 Introduction

The introduction of Transformer-based architectures [1] has opened new opportunities for integrating high-level semantic reasoning with low-level geometric navigation in robotics. Traditional robotic exploration methods have primarily focused on mapping unknown environments using geometric cues, often neglecting the rich semantic information available in visual and linguistic modalities [2]. However, recent advances in Large Language Models (LLMs) [3] and Vision-Language Models (VLMs) [4] have enabled robots to interpret and act upon complex, open-ended instructions expressed in natural language [5]. These developments mark a transition from deterministically modeled and explicitly programmed robotic systems toward zero-shot generalizable behavior, enabling robots to reason about previously unseen concepts beyond fixed, task-specific datasets.

Consequently, new applications have emerged that require robots not only to explore and map their surroundings but also to reason about the semantic structure and relationships within them. For instance, in sevice robotics a mobile agent may be instructed to locate a specific object based on high-level descriptions such as *"find the red chair in the living room"*, rather than relying on a limited set of predefined categories such as those from Common Objects in Context (COCO) [6]. Similarly, in Search and Rescue (SAR) operations [7], robots may be tasked with locating missing persons based on vague or incomplete contextual information, such as the assumption that an individual might be found *"in the bathroom"*. In industrial inspection, autonomous agents must identify structural anomalies or specific components within unstructured and partially observable environments, while in warehouse automation, robots must locate items or storage units that may not be consistently labeled or fully visible.

Across these domains, the integration of semantic understanding with autonomous navigation is an active area of research [8–11]. Robots must be capable of interpreting abstract human instructions. Traditional geometric exploration approaches [2, 12, 13] focus on exploring unknown environments by maximizing information gain about the spatial structure. However, in semantic exploration tasks, the robot must reason jointly about spatial and semantic context to strategically locate target objects or regions of interest [9, 10, 14, 15].

Current research increasingly leverages pretrained VLMs to extract semantic cues from RGB images, enabling zero-shot reasoning about novel objects and scenes [8, 9, 11]. However, many of these approaches rely on short-term or episodic semantic representations and lack mechanisms for persistent spatial memory or principled integration of semantic information into long-term exploration decisions. Furthermore, the dynamic and partially known nature of real-world environments necessitates efficient search strategies that balance exploration of unknown areas with exploitation of previously acquired knowledge [16–19].

These challenges motivate the development of a unified framework that bridges geometric exploration with semantic scene understanding by balancing semantic frontier-based ex-

ploration with long-term semantic memory, enabling autonomous agents to perform open-vocabulary, goal-directed exploration guided by high-level semantic input.

1.1 Language-Guided Exploration

Traditional geometric exploration techniques are widely used for mapping unknown environments by identifying frontiers in either two or three dimensions and navigating toward the frontier with the highest expected information gain [13, 20]. Such methods, including those based on occupancy grids [21] or point cloud representations [22], are particularly effective for coverage and mapping tasks and have been successfully extended to multi-robot systems for large-scale exploration [23]. However, these approaches remain primarily geometry-driven and do not incorporate semantic understanding of the environment. Consequently, they are suboptimal for goal-directed exploration tasks, where the objective is to locate specific objects or regions of interest defined by high-level semantic criteria rather than unexplored geometry [15].

To address this limitation, recent research has focused on integrating semantic perception into exploration frameworks [8–11, 14, 24–26]. Instead of relying solely on LiDAR or depth sensors for geometric mapping, the use of RGB imagery enables semantic reasoning about the scene and the objects contained within it. Table 1 summarizes representative works that leverage either pretrained VLMs or learning-based navigation policies trained via Behavioral Cloning (BC) or Reinforcement Learning (RL), to guide robots toward regions that are semantically relevant to a given target description in natural language.

Approach	Training Required	Real-Time	Semantic Reasoning Model
VLFM [9]	✗ (zero-shot)	✓	BLIP-2 [27] + GroundingDINO [28] + SAM [29]
SemUtil [8]	✗ (training-free)	✓	Mask R-CNN [30] + CLIP [4] + BERT [31]
ESC [10]	✗ (zero-shot)	✓	GLIP [32] + DeBERTa [31] / ChatGPT reasoning
LGX [24]	✗ (zero-shot)	✓	GPT-3 [3] + GLIP [32] + BLIP-2 [27]
CoW [14]	✗ (zero-shot)	✓	CLIP similarity scoring [4]
ZSON [11]	✓ (Reinforcement Learning pretraining)	✗	CLIP-based RL policy
PONI [25]	✓ (supervised)	✗	Learned potential-field network
PIRLNav [26]	✓ (Behavior Cloning + Reinforcement Learning)	✗	DINO-based CNN-RNN policy [33]

Table 1: Overview of semantic zero-shot and trained exploration approaches. All of these methods are capable of navigating to a goal described in natural language, however, they differ in zero-shot applicability to new scenes and real-time capability.

Approaches such as ZSON [11], PONI [25], and PIRLNav [26] employ deep reinforcement

learning (DRL) or supervised training to develop navigation policies capable of generalizing to unseen objects. Although these methods achieve promising results in simulation, they require extensive offline training and exhibit limited adaptability to previously unseen environments or object categories not encountered during training. In contrast, zero-shot methods such as Vision-Language Frontier Maps (VLFM) [9], SemUtil [8], ESC [10], LGX [24], and CoW [14] leverage pretrained VLMs to perform semantic exploration without additional training. These models enable real-time decision-making by exploiting semantic cues extracted from RGB imagery, guiding robots toward areas likely to contain the target object or region.

Some approaches, such as ESC [10] and LGX [24], further integrate LLMs for commonsense reasoning and high-level task interpretation, enabling a more contextual understanding of complex instructions. However, this comes at the cost of increased computational demand and potential inference latency, which limits their applicability on resource-constrained mobile platforms. While VLFM [9] achieves real-time performance, it relies on multiple computationally expensive foundation models (GroundingDINO [28], Segment Anything Model (SAM) [29], and Bootstrapped Language Image Pretraining 2 (BLIP-2) [27]) that require high-end GPUs with up to 16 GB of Video Random Access Memory (VRAM), which is impractical for embedded robotic systems.

Overall, these language-guided exploration methods primarily focus on short-term semantic reasoning and lack persistent memory. They do not maintain long-term storage or recall mechanisms for previously acquired semantic knowledge, leading to redundant exploration and reduced efficiency in multi-object and chained search tasks.

1.2 Language-Embedded Semantic Mapping

A language-embedded semantic map serves as a persistent spatial memory that jointly encodes the geometric structure of the environment and its semantic content. Semantic information can be incorporated either by associating discrete object classes [34], inferred by a visual perception backbone, with their spatial locations, or by embedding high-dimensional visual representations into a spatial map [17, 35, 36], such as a projected voxel grid. In the latter case, the visual embeddings capture semantic properties of objects and regions beyond fixed category labels [9, 17, 35]. This abstraction allows semantic information to be reused across tasks and time horizons, rather than being tied to a single perception or navigation episode [16].

Such semantic representations can be queried using natural language prompts, enabling robots to reason about the presence, distribution, and spatial relationships of objects or regions of interest based on high-level descriptions [10, 15]. By leveraging either object class information or continuous visual embeddings, a robot can guide its navigation toward areas with a high likelihood of containing a specified target [16, 18], thereby improving search efficiency and task success rates. Furthermore, language-embedded semantic maps can be combined with high-level reasoning modules, such as Large Language Models (LLMs), to infer object relationships, contextual cues, and action sequences required to accomplish more complex, multi-step goals [10, 24].

Maintaining such semantic representations persistently over time enables robots to exploit

past observations, recall previously detected objects, and avoid redundant exploration of already known regions. This form of long-term global memory improves navigation efficiency, scalability, and robustness in open-vocabulary, real-world environments [16, 19]. Table 2 summarizes representative works that incorporate persistent or memory-based semantic mapping to enhance exploration capabilities.

Approach	Training Required	Real-Time	Memory Representation	Exploration Integration
OneMap [16]	✗ (zero-shot)	✓	2D probabilistic feature field	✓ (frontier-based)
ConceptGraphs [17]	✗ (pretrained models)	✗	3D scene graph	✓ (LLM-planner)
SemExp [34]	✓ (RL + supervised)	✗	2D semantic occupancy map	✓ (learned policy)
GeFF [37]	✓ (ScanNet pretrain)	✓	Implicit 3D feature field	✗ (passive)
RayFronts [15]	✗ (foundation model)	✓	Hybrid voxel + ray field	✗ (planner-agnostic)
VLMaps [18]	✗ (pretrained LSeg/CLIP)	✗	2.5D open-vocab grid	✓ (frontier-compatible)
Pigeon [19]	✓ (RLVR fine-tune)	✓	Point-of-Interest snapshot memory	✓ (reasoning-aware)

Table 2: This table provides an overview of representative persistent or memory-based semantic mapping approaches, comparing their training requirements, real-time capability, underlying memory representations, and the extent to which semantic memory is integrated into exploration or planning.

methods such as SemExp [34], Pigeon [19], and GeFF [37] rely on offline training to construct persistent semantic representations. In the case of policy-learning approaches, this dependence on extensive training limits adaptability to previously seen environments and unseen object categories. Other approaches, such as Open-Vocabulary 3D Scene Graphs (ConceptGraphs) [17] and Visual Language Maps (VLMaps) [18], construct persistent open-vocabulary maps using pretrained foundation models but often require pre-mapping and lack real-time performance, which restricts their use in dynamic or large-scale settings.

While One Map to Find Them All (OneMap) [16] demonstrates real-time onboard operation on embedded platforms such as the Jetson Orin AGX, its overall map update rate of approximately 2 Hz may limit applicability to high-speed navigation scenarios. Moreover, the quality of the probabilistic semantic belief map remains sensitive to depth sensing noise, as feature localization uncertainty increases with distance, directly affecting the spatial accuracy of semantic information. Although OneMap incorporates a consensus-filtering mechanism combining CLIP-based similarity maps with open-vocabulary object detectors to mitigate false positives, its semantic reasoning ultimately relies on vision-language similarity, which remains inherently ambiguous under open-vocabulary conditions.

GeFF! (**GeFF!**) [37] provides a compact implicit 3D representation by distilling CLIP-aligned

features into a neural field, enabling both geometric and semantic understanding. However, it requires pretraining on large-scale datasets such as ScanNet, limiting its direct generalization to arbitrary environments. RayFronts [15] introduces a hybrid 3D representation that combines voxel-based semantics with ray-based frontier expansion, offering real-time operation and high efficiency for open-set semantic search. Nevertheless, its computational complexity grows with environment size, and its planner-agnostic design prevents it from actively guiding exploration toward semantically relevant regions.

Table 3 summarizes the key limitations observed across existing semantic exploration frameworks. These gaps illustrate the need for a unified approach that combines zero-shot semantic understanding, persistent spatial memory, and real-time exploration to achieve robust, scalable autonomy in complex environments.

Limitation	Example Works	Implication
No persistent memory	VLFM [9], CoW [14], LGX [24], ESC [10]	No long-term fusion or recall; repeated exploration of known areas.
Offline training required	ZSON [11], PONI [25], PIRLNav [26], SemExp [34]	Heavy RL/supervised training; poor adaptability to new scenes.
No explicit exploration-memory trade-off	OneMap [16], RayFronts [15], VLMaps [18]	Either passive mapping or short-term exploration; inefficient search.
No zero-shot exploration	CoW [14], LGX [24]	Detect novel objects but fail to explore unseen regions strategically.
Premapping needed	ConceptGraphs [17], **GeFF!** [37]	Depend on pre-recorded data; not suited for online autonomy.
Limited robustness	PONI [25], **SemExp!** (**SemExp!**) [34], PIRLNav [26]	Closed-set categories; fragile under real-world variation.
Low real-world applicability	ConceptGraphs [17], VLMaps [18]	High Graphics Processing Unit (GPU) cost or simulation-only evaluation; limited deployability on mobile robots.

Table 3: This table summarizes key limitations of existing semantic exploration frameworks, providing representative example works for each limitation and outlining their practical implications for autonomous navigation and exploration.

These observations reveal several fundamental challenges that are not yet adequately addressed by existing semantic exploration frameworks.

First, many approaches lack the ability to perform zero-shot exploration while maintaining a persistent and incrementally updated semantic representation of the environment [8–11, 14, 25, 26]. As a result, semantic information is often either discarded after individual navigation episodes or requires pre-mapped environments, limiting applicability in unknown or changing scenes.

Second, a strong reliance on computationally expensive deep reinforcement learning or supervised training pipelines remains common. This dependence restricts adaptability to new environments and object categories and poses significant challenges for deployment on resource-constrained robotic platforms [11, 14, 25].

Third, existing methods struggle to robustly handle the reliability of semantic information during exploration. Semantic maps constructed from open-vocabulary perception are inherently noisy and uncertain [4, 38], and indiscriminate reliance on semantic memory can lead to inefficient navigation [18], as robots may pursue low-confidence or spurious cues instead of exploring informative regions. In such cases, poorly calibrated use of semantic memory may negate the potential benefits of semantic guidance [25].

Conversely, approaches that rely exclusively on semantic-driven exploration may overlook structurally plausible regions suggested by prior observations, resulting in reduced task success. This challenge is further amplified in dynamic environments, where previously stored semantic information may become outdated or misleading over time.

Fourth, real-world deployment introduces additional challenges related to computational efficiency, robustness to sensor noise, and environmental variability. Many current systems struggle to sustain reliable real-time performance under embedded hardware constraints, leading to increased latency or unstable inference behavior [9, 17].

Finally, several existing approaches rely on single-source detection or similarity pipelines, which are prone to false positives and semantic ambiguities under open-vocabulary conditions. The lack of multi-source semantic validation and memory-aware reasoning limits robustness and consistency during long-term autonomous operation [9, 14, 16, 24].

1.3 Scientific Contribution

This work contributes to the state of the art by introducing a hybrid semantic exploration framework that integrates zero-shot semantic frontier scoring with persistent 3D scene representation, enabling autonomous robotic search guided by open-vocabulary text queries. The proposed method combines real-time semantic reasoning during exploration with a long-term spatial memory, allowing the robot to dynamically balance between discovering new information and exploiting previously acquired knowledge.

Unlike previous approaches that focus exclusively on either geometric frontiers or static semantic maps, the proposed framework continuously fuses information from multiple semantic sources to maintain a unified, confidence-based semantic world representation. Tunable weighting enables the robot to adjust its behavior between exploration and exploitation according to the reliability of recent observations and the stability of stored semantic memory.

The proposed method is evaluated with respect to how the quality and granularity of the underlying semantic information influence task success, navigation efficiency, and robustness. By systematically varying the trust between exploration and memory components, this thesis provides new insights into how semantic reasoning and persistent mapping can be effectively combined for open-vocabulary, multi-object search in dynamic environments.

To evaluate the contribution of the proposed method, the following research questions are formulated:

1. **How does integrating zero-shot semantic exploration and persistent 3D semantic mapping affect multi-object search performance and navigation efficiency compared to existing methods?**
   Performance is quantified with respect to task success and path efficiency, measured through Success Rate (SR), Success weighted by Path Length (SPL), and Multi-Object Success Rate (MSR) relative to representative state-of-the-art systems such as OneMap [16], VLFM [9], and Pigeon [19].

2. **How does the interaction between live exploration and accumulated semantic memory influence overall system performance?**
   The weighting factor between exploration and memory is varied during graph node fusion to assess impacts on SR and SPL, identifying optimal trade-offs between reactivity and exploitation.

3. **How robust is semantic navigation to variations in semantic memory granularity, and can exploration reduce the negative effects of coarse semantic representations?**
   Navigation performance is evaluated across different levels of semantic retrieval granularity and compared balanced exploration against pure exploitation to assess robustness using SR and SPL.

4. **How does multi-source fusion of detection confidence, semantic similarity, and memory confidence impact detection robustness and false-positive suppression during exploration?**
   This question is evaluated by analyzing Precision, Recall, F1-Score, the Confusion Matrix, and Success Rate (SR) under different fusion weight configurations across COCO, open-vocabulary, and zero-shot object classes.

5. **What is the computational footprint and real-world robustness of the hybrid framework?**
   This aspect is assessed using Frames Per Second (FPS), GPU and Central Processing Unit (CPU) utilization, inference latency, and detection stability under sensor noise during physical deployment on a mobile robot.

1.4 Thesis Structure

This work is structured as follows. Chapter 2 describes the state-of-the-art in frontier-based exploration, Reinforcement Learning (RL)-based exploration, foundation-model-based exploration and persistent semantic mapping approaches. Chapter 3 describes the methods used, for hybrid semantic exploration, persistent 3D mapping, promptable zero-shot detection, and multi-source fusion strategies. Chapter 4 details the practical implementation of the proposed

approach, covering the simulation and real-world setup, datasets, software stack, and hardware configuration. Chapter 5 presents the experimental evaluation, including ablation studies, comparative benchmarks, and real-world validation. Finally, Chapter 6 concludes the thesis with a summary of findings, discussion of limitations, and suggestions for future research directions.

2 State of the Art

This work introduces a hybrid semantic exploration framework that combines open-vocabulary semantic perception with autonomous exploration and persistent spatial memory. The proposed approach integrates concepts from geometric exploration, vision-language-guided exploration, and semantic mapping. It is motivated by recent progress in incorporating semantic information from large pretrained foundation models into classical exploration and mapping pipelines. However, these novel semantic methods still rely on design and data representation patterns derived from classic geometric exploration and mapping methods.

Semantic exploration approaches differ fundamentally in how exploration behavior is generated. Some methods learn end-to-end navigation policies using reinforcement learning or imitation learning, where exploration strategies are implicitly encoded in a trained policy optimized via task-specific reward functions [11, 26]. Other approaches adopt modular architectures that integrate pretrained vision-language models with classical mapping and planning techniques [9, 14, 16].

Reinforcement learning and imitation learning approaches have demonstrated promising results in semantic object search tasks by training agents to navigate toward target objects based on high-level semantic cues [11], primarily within simulated environments. Their main advantage lies in avoiding hand-designed exploration heuristics, as complex behaviors can be learned directly from data through reward optimization [39]. However, such policies typically require extensive training on large datasets (e.g. Replica [40], Habitat [41]), exhibit limited generalization to unseen environments or object categories [11, 26], and lack interpretability due to their black-box nature [39].

In contrast, modular approaches leverage semantic representations provided by large-scale pretrained VLMs to guide exploration decisions without task-specific retraining [15]. By combining explicit geometric mapping with semantic reasoning derived from foundation models, these systems can achieve zero-shot generalization to novel objects and environments while retaining interpretability and adaptability [9]. Several works extend this paradigm by constructing persistent semantic maps that retain knowledge of previously explored areas [16, 18], enabling more efficient multi-object search and the integration of high-level natural language instructions [17].

Finally, object detection and promptable vision-language models play a crucial role in enabling open-vocabulary semantic understanding for exploration tasks. Recent advances in grounding-capable detectors and segmentation models facilitate zero-shot object recognition based on text prompts, allowing robots to identify and localize previously unseen objects [28, 32, 42]. These models form the semantic foundation upon which hybrid exploration systems are built, enabling flexible object search in diverse real-world environments.

Frontier-based exploration selects navigation goals from geometric map structure (fron-

tiers) to maximize coverage and information gain, independent of any task-specific training. Reinforcement-learning-based semantic exploration learns a navigation policy from simulated experience (reinforcement or imitation learning) that maps observations and a goal specification to actions, implicitly encoding exploration and target-seeking behavior. Foundation-model-based semantic exploration uses pretrained vision-language foundation models as semantic priors to score observations or candidate goals by relevance to a text query, enabling zero-shot generalization within a modular planning pipeline.

2.1 Frontier-Based Exploration

Frontier-based exploration is a standard paradigm for autonomous exploration that selects goal locations on the boundary between explored free space and unknown areas to expand the map [12]. A *frontier* is defined as the boundary between known free space and unknown regions of the environment [12]. Frontier-based exploration relies on the principle that unexplored areas adjacent to known free regions provide the highest potential information gain. To identify such frontiers, the robot must maintain a global representation of the environment, typically an occupancy grid or voxel map, where each cell is classified as *free*, *occupied*, or *unknown*, based on sensor observations from Light Detection and Ranging (LiDAR) or RGB-D cameras. The robot then iteratively selects and navigates to frontiers to expand its knowledge of the environment.

Quin et al. [20] evaluated three commonly used frontier extraction methods that differ primarily in computational efficiency and scalability. The first approach, known as the *Naïve Active Area (NaïveAA)* method, evaluates every cell in the occupancy grid to determine whether it is free and has at least one unknown neighbor. Although this approach is conceptually simple and accurate for small-scale maps, it becomes computationally expensive for larger environments and often produces small, fragmented frontier clusters.

The second approach, the *Wavefront Frontier Detector (WFD)* [2, 20], improves efficiency by using a breadth-first search (BFS) to identify connected frontier regions without exhaustively scanning the entire map. Unlike the NaïveAA method, WFD directly extracts continuous frontier clusters rather than treating each frontier cell individually, significantly reducing redundant computations.

The third method, the *Frontier-Tracing Frontier Detection (FTFD)* [20], further enhances performance by incrementally updating frontier information using only the most recent sensor observations. Instead of re-evaluating the full map, FTFD initiates a BFS from previously known frontier cells that remain within the active area and from the endpoints of the latest sensor rays. Newly visible free-space cells along the scan boundary are evaluated as potential frontiers, while outdated frontier cells that are now occupied or re-observed are removed. By restricting computation to the local scan perimeter, FTFD achieves significantly faster update rates than NaïveAA and WFD, supporting real-time frontier detection even in large-scale environments.

After frontiers have been extracted, a selection strategy determines which frontier the robot should explore next. Simple heuristics such as *nearest-frontier selection* minimize travel distance but can lead to oscillatory behavior between nearby frontiers. Alternatively, selecting the

largest frontier favors unexplored regions of higher spatial extent, reducing dead-end visits but increasing traversal cost. To address these trade-offs, Bourgault et al. [13] introduced a *utility-based frontier selection* framework that combines multiple criteria, such as distance, frontier size, and expected information gain, into a unified objective function. This approach enables more balanced decision-making, improving overall exploration efficiency and map completeness. Many subsequent works have built upon this foundation, incorporating additional factors such as energy consumption, obstacle density, and dynamic environment considerations into the utility function [12].

However, all these methods are primarily designed for geometric exploration without incorporating semantic understanding. As a result, they are optimized for complete map coverage rather than goal-directed exploration tasks. In scenarios where a robot must locate specific objects or regions based on semantic cues, purely geometric frontier selection often leads to inefficient search behavior and unnecessary traversal. This motivates the integration of semantic reasoning into the frontier-based exploration process, where frontiers can be prioritized not only by geometric utility but also by their semantic relevance to the task objective.

2.2 RL-based Semantic Exploration

In contrast to geometric exploration, which aims to maximize map coverage using the shortest possible path and time, semantic exploration focuses on efficiently locating specific objects or regions of interest described in high-level semantic terms. The objective is to minimize path length and exploration time while prioritizing areas likely to contain relevant targets rather than achieving complete spatial coverage.

Semantic exploration approaches differ fundamentally in how exploration behavior is generated and optimized. RL and imitation learning methods have demonstrated promising results by training agents to navigate toward target objects based on high-level semantic cues [11, 25, 26]. In these approaches, an agent learns a policy that maps high-dimensional sensory observations to low-level control actions by optimizing a task-specific reward function, rather than relying on explicitly designed exploration heuristics [39].

In the context of semantic exploration, RL-based methods are typically formulated as Markov Decision Processes or, more commonly in embodied navigation, as partially observable Markov Decision Processes (Partially Observable Markov Decision Processs (POMDPs)) [21]. At an abstract level, a POMDP can be defined as a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where $\mathcal{S}$ denotes the latent environment states, $\mathcal{A}$ the set of possible actions (e.g., move forward, turn left/right), $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ the state transition probability function, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward function, and $\gamma \in [0, 1)$ the discount factor. The agent does not observe the full state directly but instead receives high-dimensional, partial observations (e.g., RGB images, depth measurements, or semantic representations), from which it must infer an internal belief about the environment [39].

The reward function encodes task objectives such as reaching a target object, minimizing path length, or maintaining correct orientation, thereby implicitly shaping the agent's exploration behavior [39]. This paradigm has been successfully applied to semantic object-goal navigation,

where agents are trained to locate objects specified by high-level semantic descriptions (e.g., object categories or language embeddings) [11, 25, 26].

A key advantage of RL-based semantic exploration lies in its flexibility. Complex navigation behaviors can be learned directly from interaction data without manually designing exploration strategies. However, this flexibility comes at the cost of extensive training requirements, limited interpretability, and reduced robustness to domain shifts, as policies often overfit to the visual statistics and dynamics of the training environments [11, 39]. As a result, many such approaches are primarily evaluated in simulation and struggle to apply to previously unseen scenes, object appearances, or sensor configurations. Commonly, RL algorithms are optimized via policy gradient methods [43], which directly adjust the policy parameters θ to maximize the expected cumulative reward $J(\theta)$:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid o_t)\, R(\tau) \right], \tag{1}$$

where τ denotes trajectories sampled from the policy π_θ, a_t and o_t are the action and observation at time t, respectively, and $R(\tau)$ is the cumulative reward obtained over trajectory τ.

Majumdar et al. [11] proposed Zero-Shot Object Navigation (ZSON), a zero-shot object navigation framework that leverages the shared embedding space of CLIP [4] to guide navigation policies trained via RL. During training, the agent observes RGB images and previous actions, while the navigation goal is specified by the CLIP embedding of an image containing the target object. The policy is optimized using a shaped reward function:

$$r_t = r_{\text{success}} + r_{\text{angle-success}} - \Delta d_{tg} - \Delta a_{tg} + r_{\text{slack}}, \tag{2}$$

where r_{success} rewards successful target localization, $r_{\text{angle-success}}$ encourages correct orientation, Δd_{tg} and Δa_{tg} penalize distance and angular deviation, respectively, and r_{slack} is a negative constant, which penalizes inefficient actions per step [11]. At inference time, the image-based goal embedding is replaced by the CLIP embedding of a textual object description, enabling zero-shot generalization to unseen object categories. The action space is discrete and consists of *move forward*, *turn left*, *turn right*, and *stop*. The reward function is used exclusively during training to learn the navigation policy during inference, the agent selects actions solely based on the learned policy without access to the reward signal.

While ZSON demonstrates strong zero-shot object generalization, exploration behavior remains implicitly encoded in the learned policy, which makes it difficult to interpret or adapt to new scenarios. As a consequence, the agent tends to revisit visually familiar regions and lacks explicit mechanisms for systematic exploration of unknown space. Furthermore, the end-to-end RL formulation reduces interpretability and necessitates retraining when visual conditions or environment layouts deviate from the training distribution. Nevertheless, ZSON represents a significant step toward flexible and generalizable semantic exploration by integrating vision-language models with reinforcement learning.

Ramrakhya et al. [26] introduced Pretraining with Imitation and Reinforcement Learning (PIRLNav), a two-stage framework that combines Behavioral Cloning (BC) with RL to improve

generalization in semantic navigation tasks. In the first stage, a navigation policy is pretrained via imitation learning on large-scale human demonstrations, learning to reproduce expert actions from observations that include Red Green Blue (RGB) images, pose information, and a categorical goal representation. This pretraining stage provides a strong initialization, which reduces the training time and sample complexity required for subsequent reinforcement learning. The behavior cloning objective is formulated as a cross-entropy loss over expert actions (see Equation 3).

A stochastic navigation policy is denoted by $\pi_\theta(a \mid s)$, parameterized by θ, mapping an observation s to a distribution over actions a. Training objectives are expressed as loss functions $\mathcal{L}(\theta)$ minimized with respect to θ. The pretrained ObjectNav policy is trained using supervised learning to minimize the behavior cloning loss in Equation 3,

$$\mathcal{L}_{\text{BC}}(\theta) = -\sum_{t=1}^{T} \log \pi_\theta(a_t^* \mid s_t), \tag{3}$$

where a_t^* denotes the expert action at time step t and s_t denotes the agent observation. The ObjectNav policy is trained with a Convolutional Neural Network (CNN)+Recurrent Neural Network (RNN) architecture, in which a pretrained vision backbone, Self-Distillation with No Labels (DINO), extracts visual features from RGB images and feeds them into the navigation policy network along with pose and goal information.

In the second stage, the pretrained policy is fine-tuned using RL by maximizing the expected discounted return (see Equation 4):

$$\theta^* = \arg\max_\theta \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=1}^{T} \gamma^{t-1} r_t \right], \tag{4}$$

where $\tau = (s_1, a_1, \ldots, s_T, a_T)$ denotes a trajectory generated by executing π_θ, r_t is the reward at time step t, and $\gamma \in (0, 1]$ is the discount factor. Although this hybrid training strategy improves sample efficiency and navigation robustness in simulation, the resulting policy remains a black-box model. It does not maintain an explicit semantic memory and requires retraining to adapt to new visual domains or sensor modalities.

To address the limited interpretability of end-to-end policies, Ramakrishnan et al. [25] proposed Potential Functions for Object-Goal Navigation (PONI), a supervised, map-based semantic exploration framework. Rather than learning low-level actions, PONI predicts high-level exploration objectives in the form of potential fields defined over a partial semantic map. Two complementary potentials are estimated: an area potential that encourages exploration of unknown space U_t^a, and an object potential that estimates proximity to the target category U_t^o. Exploration decisions are derived by scoring geometric frontiers according to a weighted combination of these potentials (see Equation 5),

$$U_t(f) = \alpha\, U_t^a(f) + (1 - \alpha)\, U_t^o(f) \tag{5}$$

where α explicitly controls the trade-off between exploration and exploitation. This formulation yields interpretable and stable exploration behavior and decouples high-level decision-making from low-level navigation, which is handled by a classical navigation planner.

However, PONI relies on dense semantic annotations and assumes a fixed, closed set of object categories encountered during training. [25] used Mask R-CNN [30] to generate semantic maps with 80 object classes from the COCO dataset [6], which was finetuned within the Gibson dataset [44]. As a result, it does not support open-vocabulary or zero-shot object search and remains sensitive to annotation noise and domain shifts.

RL-based and supervised semantic exploration methods demonstrate the feasibility of learning navigation behavior from semantic cues, but they exhibit recurring limitations that motivate alternative approaches. These include extensive training requirements, limited interpretability, closed-set semantics, lack of persistent semantic memory, and reduced robustness to domain changes. These structural shortcomings have motivated recent work toward modular exploration frameworks that integrate pretrained vision-language models, which are discussed in the following section.

2.3 Foundation-Model-Based Semantic Exploration

In contrast to reinforcement learning-based approaches, which derive navigation behavior through task-specific training, a second line of work leverages large-scale pretrained VLMs as semantic priors within modular exploration systems. These approaches shift the learning burden away from navigation policy optimization toward semantic perception and reasoning, enabling zero-shot generalization to novel objects and environments without task-specific retraining.

VLMs are large pretrained image-text models that learn joint representations of visual and linguistic data from massive web-scale datasets [4, 32]. Their core principle is to embed images and text into a shared latent space, in which semantically related visual and linguistic concepts are mapped to nearby representations. A VLM defines two embedding functions, $f_I(\cdot)$ and $f_T(\cdot)$, which map an image I and a text prompt T to a common embedding space $\mathbb{R}^d$:

$$\mathbf{e}_I = f_I(I), \qquad \mathbf{e}_T = f_T(T), \tag{6}$$

where $\mathbf{e}_I, \mathbf{e}_T \in \mathbb{R}^d$ are high-dimensional feature vectors encoding semantic information. Text inputs are tokenized and processed using transformer-based language encoders, while visual inputs are decomposed into patches or regions and encoded by a vision backbone (e.g., CNN or transformer-based architectures), depending on the model design [1, 27, 32]. Semantic alignment between image and text embeddings is commonly quantified using cosine similarity, which measures the angular similarity between vectors in the shared embedding space (see Equation 7),

$$\mathrm{sim}(I, T) = \frac{\mathbf{e}_I \cdot \mathbf{e}_T}{\|\mathbf{e}_I\| \, \|\mathbf{e}_T\|} \tag{7}$$

A higher similarity score indicates stronger semantic correspondence between the visual observation and the textual query. This representation enables open-vocabulary reasoning, as arbitrary object descriptions can be matched against visual observations without retraining, forming the foundation for zero-shot semantic perception in exploration tasks.

Foundation-model-based exploration is characterised by explicit separation between geometric navigation and semantic understanding. Geometric structure is typically handled by classical mapping and planning components (e.g., frontier-based exploration), while semantic relevance is inferred from pretrained models such as CLIP [4], BLIP-2 [27], or grounding-capable detectors [9, 14, 28]. This modularity improves interpretability and adaptability, but introduces new challenges related to uncertainty handling, semantic consistency, and long-term memory.

Table 4 summarizes foundation-model-based exploration frameworks in five categories: (1) the source of semantic signals, (2) the mechanism used to fuse semantics with geometric exploration, (3) the handling of detection confidence and uncertainty, (4) the underlying semantic data representation, and (5) the dominant failure causalities observed in practice.

Method	**Semantic signal source**	**Fusion with geometry**	**Detection confidence handling**	**Semantic representation**	**Primary failure causality**
ESC [10]	Object detections + LLM priors	Probabilistic frontier scoring (PSL)	Single-step confidence, no revision	Local symbolic semantic map	False positives amplified by reasoning priors
CoW [14]	Image-text similarity CLIP	No explicit geometric fusion	Threshold-based similarity	No explicit map	Oscillation and local minima near false positives
SemUtil [8]	Closed-set detections + CLIP similarity	Utility map over geometric frontiers	No confidence decay or belief update	Semantic point cloud + scene graph	Persistent corruption from misdetections
VLFM [9]	Dense image-text similarity BLIP	Semantic value-map fused with frontier map	Weighted averaging over observations	Episodic semantic value map	False positives persist across episode

Table 4: Design patterns and limitations of foundation-model-based semantic exploration frameworks. The table compares how different methods obtain semantic signals, integrate them with geometric exploration, handle uncertainty over time, and represent semantic information, revealing recurring failure modes that motivate the need for persistent semantic memory and belief revision.

A representative example of this paradigm is the Exploration with Semantic Cues (ESC) framework proposed by Zhou et al. [10], which augments traditional frontier-based exploration with semantic cues derived from pretrained VLMs. Specifically, ESC combines a grounded object detector, **GLIP!** (**GLIP!**) [32], with a LLM (either ChatGPT or DeBERTa) to generate semantic priors that guide exploration decisions. Frontiers are scored based on both geometric utility and semantic relevance to the target object (see Equation 8).

$$P(F) = P(F \mid d_i^t, o^t, r^t) \tag{8}$$

$P(F)$ denotes the probability of selecting frontier F, conditioned on detected objects d_i^t, current image observations o^t, and the robot pose r^t at time t. This formulation is implemented

using Probabilistic Soft Logic (PSL), which fuses visual detections with language-derived priors about object co-occurrences and spatial relationships. The robot then selects the frontier with the highest combined score, balancing geometric and semantic information to improve search efficiency. For navigating toward target objects, a classical A-star (A*) planner is employed to compute collision-free paths based on the occupancy map.

Zhou et al. [10] employs **GLIP!** [32] as the detection backbone to compute 2D bounding boxes, class labels, and confidence scores for objects within the robot's Field of View (FOV). While effective in simulation, the approach introduces notable computational overhead due to repeated LLM inference and PSL optimization. As a consequence of relying on single-step detections and static commonsense priors, errors introduced by false positives are not attenuated over time. Once a misleading semantic hypothesis is introduced, the probabilistic reasoning layer tends to reinforce rather than correct it, leading to persistent semantic bias during exploration.

Through ablation studies, Zhou et al. [10] observed that object-object and object-room relational priors can occasionally degrade performance, as commonsense relationships are inherently probabilistic rather than deterministic. Additionally, while ESC maintains a local semantic map during navigation, it lacks mechanisms for long-term memory or belief revision. Consequently, once an incorrect detection or prior is introduced, the framework has no learned means of down-weighting or correcting it over time, which can lead to persistent semantic inconsistencies during extended exploration.

In contrast to such LLM reasoning-based systems, Gadre et al. [14] proposed CLIP on Wheels (CoW), a lightweight vision-language exploration framework that relies purely on image-text alignment from CLIP [4] without requiring explicit frontier detection, semantic mapping, or object segmentation. The method guides the robot toward directions with the highest cosine similarity (see Equation 7) between the current visual observation and the target object description.

By eliminating explicit detectors and handcrafted mapping, CoW offers a computationally efficient and conceptually simple baseline for open-vocabulary navigation. However, this simplicity comes at the cost of robustness. The system is highly sensitive to viewpoint variations and clutter, as cosine similarity does not always correlate with true object presence. Without spatial memory or geometric reasoning, the robot may oscillate near false positives or become trapped in local minima. Moreover, because similarity scores vary across object categories, no universal threshold can be established for all targets, resulting in inconsistent stopping behavior and reduced reliability during multi-object search.

Building upon this idea of integrating semantics into classical exploration, Chen et al. [8] introduced Semantic Utility Maps (SemUtil), a fully modular and training-free framework for object-goal navigation that combines classical SLAM-based mapping with pretrained perception and language models. In contrast to reinforcement learning or imitation learning approaches, SemUtil leverages explicit geometric and semantic reasoning through three core components: a 2D occupancy map for frontier extraction, a semantic point cloud generated by projecting Mask R-CNN detections into 3D space, and a spatial scene graph for high-level semantic reasoning. These three representations collectively form a structured scene model

that supports geometric planning, semantic propagation, and reasoning about unexplored regions [8].

The central element of SemUtil is the *utility module*, which fuses geometric frontiers with semantic priors to determine the most promising frontier to explore next. For each map cell, a *utility score* is computed by combining the geometric frontier characteristics, the CLIP-based cosine similarity between the current observation and the target object description, and the semantic cues from the 3D point cloud (e.g., class IDs from Mask R-CNN). This results in a utility map that prioritizes frontiers both spatially and semantically, as illustrated in Figure 3 of the original paper (showing the interaction between geometric and semantic utilities) [8]. SemUtil solves the oscillation issues observed in CoW by explicitly extracting frontiers from the occupancy map and scoring them based on their semantic utility, rather than relying solely on raw similarity scores. This structured approach enables more stable exploration behavior and reduces the likelihood of becoming trapped near false positives, which also applies to other works, which combine VLMs with frontier-based exploration [9].

Importantly, the utility map in SemUtil is not persistent, it is recomputed at every timestep based solely on the current observation and semantic point cloud, without maintaining a long-term memory of past detections or map updates. While this design simplifies computation and eliminates the need for training, it also limits the system's ability to reason over time or correct previous errors. The reliance on a closed-set detector (Mask R-CNN) restricts open-vocabulary generalization, and any incorrect detection directly corrupts the semantic point cloud, thereby distorting the frontier scoring and leading to suboptimal exploration decisions. Furthermore, since the framework lacks belief revision or memory-based fusion, false detections persist until they leave the robot's current field of view, reducing consistency and efficiency in long-term navigation.

In contrast to SemUtil [8], VLFM [9] derives semantic values directly from RGB observations using a pretrained VLM (BLIP-2) [27]. The target relevance is the cosine similarity to the text prompt (see Equation 7) [14].

The resulting similarity values are spatially projected onto a top-down occupancy grid according to the robot's FOV, forming a *value map* that quantifies the semantic likelihood of each region leading toward the target object. To account for reduced reliability near the image periphery, a Gaussian weighting function attenuates confidence values based on angular distance from the optical axis (see Fig. 3 in the original paper [9]). This value map is continuously updated through a weighted averaging scheme that fuses new and previous similarity scores according to their confidence weights, enabling smooth map updates and spatial consistency across frames. BLIP-2 is not used for caption generation, but rather for retrieving text-image embeddings to compute similarity scores [9, 27].

During exploration, VLFM fuses this value map with a geometrically extracted frontier map to select the next exploration goal, the frontier with the highest semantic value is chosen as the next waypoint. Target object detection is performed using YOLOv7 [45] for COCO [6] categories and GroundingDINO [28] for open-vocabulary detection. Once an object matching the target query is detected, SAM [29] is applied to generate an accurate mask, and the system transitions from exploration to goal navigation.

This modular framework achieves state-of-the-art performance on the Gibson [44], HM3D, and Matterport3D [46] benchmarks, outperforming prior zero-shot approaches such as ESC, SemUtil, and CoW in both SR and SPL [9]. Despite its efficiency and interpretability, several limitations remain. VLFM relies on a single-source detection pipeline, either GroundingDINO or YOLOv7, making it prone to false positives in open-vocabulary scenarios, which can result in premature stopping behavior. Furthermore, the value map is episodic rather than persistent: it resets after each navigation episode and does not maintain long-term semantic memory, leading to redundant revisits during multi-object search tasks. While the system operates in real time, the use of multiple large-scale pretrained models (BLIP-2 [27], GroundingDINO/YOLOv7 [28, 45], and SAM [29]) demands substantial computational resources, consuming approximately 16 GB of VRAM on an NVIDIA RTX 4090 GPU during deployment, which limits scalability on embedded robotic platforms.

Foundation-model-based exploration approaches enable zero-shot semantic navigation without the need for task-specific training and offer improved interpretability compared to learned policies. However, across all reviewed methods, semantic information is either transient, episodic, or locally scoped. None of the approaches maintain a persistent semantic belief that can be revised over time as new evidence is accumulated. This lack of long-term semantic memory leads to repeated exploration, sensitivity to false detections, and inconsistent behavior in multi-object search scenarios.

These recurring limitations motivate the development of exploration frameworks that combine open-vocabulary semantic perception with persistent, revisable semantic memory, which is the focus of this work.

2.4 Persistent Semantic Mapping for Exploration

Table 5 summarizes recent approaches to persistent semantic mapping for exploration tasks. The table highlights how different methods store semantic information, update it over time, and whether they support belief revision, revealing a common lack of mechanisms for correcting erroneous semantic memories.

Method	Semantic Memory Type	Spatial Representation	Open-Vocab	Update Mechanism	Belief Revision	Primary Limitation
OneMap [16]	Open-Vocabulary Belief Map (CLIP features)	2.5D top-down grid map	✓	Uncertainty-weighted accumulative fusion	✗	Irreversible belief fusion leads to semantic drift
VLMaps [18]	Dense per-cell language embeddings (CLIP-based [4])	2.5D top-down grid map	✓	Multi-view feature averaging (accumulative fusion)	✗	Irreversible feature averaging causes semantic noise and ambiguity
PIGEON [19]	Language-conditioned episodic visual memory (Point of Interest (PoI) snapshots)	2D geometric exploration map + episodic PoI memory	✓	Episodic accumulation of visual evidence (per episode)	✗	No persistent semantic belief state or belief revision mechanism
DualMap [47]	Dual semantic maps (short-term + long-term)	2D grid maps	✓	Dual-stream accumulative fusion	✗	Long-term map cannot correct early semantic errors

Table 5: Comparison of semantic memory representations for exploration. The table highlights how different methods store semantic information, update it over time, and whether they support belief revision.

A approach to building persistent open-vocabulary semantic maps during exploration is presented by Busch et al. [16] and Huang et al. [18]. Both methods construct a 2.5D top-down grid map in which semantic information is stored at the cell level in the form of language-aligned visual embeddings, enabling open-vocabulary querying via image-text similarity.

In both works, incoming RGB-D observations are processed by a vision-language model to extract dense semantic features. In OneMap, global CLIP image embeddings are projected into the map using camera intrinsics and extrinsics, while VLMaps employs language-driven semantic segmentation (LSeg) to obtain dense per-pixel language embeddings [16, 18]. These features are associated with corresponding grid cells in the top-down map by back-projecting depth pixels into 3D space and discretizing them onto the 2.5D grid representation.

As the robot explores, newly observed embeddings are fused with existing map entries in an accumulative manner. Busch et al. [16] perform uncertainty-aware recursive fusion, in which observations with higher confidence exert greater influence on the stored semantic representation, whereas Huang et al. [18] apply multi-view feature averaging without explicit uncertainty modeling. In both cases, once semantic features are integrated into the map, they are not

selectively down-weighted or removed at later time steps.

Open-vocabulary object querying is performed by comparing stored map embeddings against the embedding of a text prompt using cosine similarity (see Equation 7). An object is considered detected if the similarity score in any map cell exceeds a predefined threshold. Because this decision relies solely on similarity values rather than explicit object detection or instance verification, both approaches are sensitive to threshold selection and may produce false positives in visually cluttered or ambiguous scenes.

Importantly, neither OneMap nor VLMaps incorporates mechanisms for belief revision or error correction. Once incorrect or noisy observations are fused into the map, they persist indefinitely and can bias future exploration decisions, leading to semantic drift over time [16, 18]. Furthermore, the projection of semantic information onto a 2.5D grid discards vertical structure and instance-level geometry, limiting semantic fidelity in complex environments and preventing accurate 3D object localization for downstream tasks such as manipulation or grasp planning [17].

Peng et al. [19] introduced *PIGEON*, a VLM-driven exploration framework that replaces persistent dense semantic maps with an episodic, object-centric memory abstraction. Instead of maintaining a globally consistent semantic map, PIGEON represents the environment as a set of semantically meaningful *PoIs*, corresponding to geometrically salient observation locations enriched with visual context.

At each PoI, the robot stores a small set of RGB observations captured from multiple viewpoints [19]. Semantic evaluation is deferred to query time, where a vision-language model jointly reasons over the language query and the stored RGB observations to assess the semantic relevance of each PoI. These relevance assessments are used to guide PoI selection, while low-level navigation between PoIs is handled by a classical planner. Reinforcement learning is employed to fine-tune the VLM's PoI selection behavior, rather than to directly select navigation actions.

Formally, each PoI p_i is defined by a spatial location $\mathbf{x}_i$ and an associated set of RGB observations $\mathcal{I}_i = \{I_{i,1}, \ldots, I_{i,K}\}$ captured from different viewpoints at that location [19]. No semantic labels, object identities, or belief states are stored; semantic interpretation is performed on demand and is not consolidated into a persistent semantic world model.

In contrast to dense mapping approaches such as OneMap [16] and VLMaps [18], PIGEON does not perform accumulative fusion of semantic features into a persistent spatial representation. As a result, it is less susceptible to semantic drift caused by irreversible belief fusion, but does not support belief revision or long-term semantic consistency across multiple object-search tasks [19].

Methodologically, PIGEON occupies an intermediate position between dense semantic mapping and object-centric persistent representations such as DualMap [47] and ConceptGraphs [17]. It combines episodic visual memory with query-conditioned semantic scoring and reinforcement learning-based navigation, while deliberately avoiding persistent semantic state estimation.

Jiang et al. [47] introduced *DualMap*, an object-centric framework for online open-vocabulary exploration that explicitly separates short-term perceptual observations from long-term seman-

tic memory. Unlike dense feature-map approaches such as OneMap [16] and VLMaps [18], which store pixel-wise or cell-wise language embeddings, DualMap reasons over discrete object instances and their spatial relations.

At each timestep, objects are detected and segmented from the RGB image using YOLO-World [48]. For each segmented object, a visual embedding is computed from the cropped image using CLIP's image encoder [4]. If a textual label is available, an additional text embedding is obtained from CLIP's text encoder. The final object-level semantic representation is computed as a weighted fusion of image and text embeddings:

$$\mathbf{f}_t = \alpha\, \mathbf{f}_t^{\text{img}} + (1-\alpha)\, \mathbf{f}_t^{\text{text}}, \qquad \alpha = 0.7, \tag{9}$$

where $\mathbf{f}_t^{\text{img}}$ and $\mathbf{f}_t^{\text{text}}$ denote the image-based and text-based embeddings at time t, respectively. This object-centric representation enables open-vocabulary semantic reasoning without requiring dense per-pixel feature storage.

DualMap maintains two complementary semantic maps. The *local concrete map* stores recently observed object instances as 3D point clouds with associated semantic embeddings, enabling rapid adaptation to new observations. In contrast, the *abstract map* serves as a long-term semantic memory and stores only stable object instances, referred to as *anchors* (e.g., tables, desks, counters), which are unlikely to change location over time. Smaller or movable objects are treated as *volatile* and are not permanently stored in the abstract map [16].

Objects are added to or updated in the maps based on a combination of semantic similarity between embeddings and geometric overlap, measured via the 3D Intersection over Union (IoU) between observed point clouds. An object is updated from the local concrete map to the abstract map only when its confidence exceeds a predefined threshold, thereby balancing adaptability with long-term stability. Each anchor in the abstract map maintains a list of associated volatile objects that have been observed in its vicinity, allowing the system to reason about object co-occurrence without permanently storing potentially transient items [16].

During object-goal exploration, DualMap does not directly search the entire map for the target object. Instead, the language query is embedded using CLIP and matched against the semantic representations of anchors and their associated volatile objects. Anchors with high semantic relevance are prioritized as navigation goals, and the robot navigates toward them using a classical A* planner [49] on the occupancy map. Once the target object is detected again in the local concrete map, exploration terminates.

By separating short-term perceptual memory from long-term semantic anchors, DualMap achieves more structured and interpretable language-guided navigation than dense feature-map approaches. However, semantic information in DualMap is primarily used for exploitation of previously observed anchors, while exploration itself remains purely geometry-driven. The framework does not provide a mechanism or hyperparameter to explicitly balance semantic exploration and exploitation, as semantic reasoning is only applied after anchors have been formed [47]. As a result, DualMap favors semantic exploitation over semantic exploration, which can limit its ability to actively search for objects that have not yet been associated with existing anchors.

2.5 Semantic Scene Reconstruction

In order to incorporate persistent semantic memory into exploration systems, to build and maintain accurate 3D reconstructions of the environment enriched with semantic information. This section reviews state-of-the-art methods for semantic scene reconstruction, focusing on how different approaches represent spatial and semantic information, their ability to operate in real time, and whether they support object-centric reasoning and which foundation models they leverage. Table 6 summarizes key characteristics of recent semantic scene reconstruction methods.

Method	Representation	Foundation Model	Zero-Shot	Real-Time capability	Object-Centric
ConceptGraphs [17]	Points	OpenCLIP [50], SAM [29]	✓	✗	✓
Clio [47]	Points (dual map)	CLIP-based	✓	✓	✓
LERF [51]	NeRF	OpenCLIP [50], DINOv2 [52]	✗	✗	✗
OpenFusion [35]	region-based TSDF	SEEM [53]	✓	✓†	✓
RayFronts [15]	Voxel grid + semantic ray field	CLIP [4]	✓	✓	✗
OTAS [54]	Points	CLIP [4], SAM2 [55], DINOv2 [52]	✓	✓	✗

†Real-time refers to geometric reconstruction. Semantic inference runs asynchronously or at a lower rate.

Table 6: Overview of semantic scene reconstruction methods and their core design choices. The table compares representation, foundation model, zero-shot applicability, real-time capability, and object-centricity.

Gu et al. [17] proposed *ConceptGraphs*, and Maggio et al. [56] introduced *Clio*, two object-centric semantic mapping frameworks that maintain persistent object-level representations enriched with open-vocabulary semantic information. Both methods represent the environment in an object-centric manner by extracting objects using open-vocabulary object detectors (e.g., GroundingDINO [28]) and segmenting them with SAM [29], yielding a bounding box and segmentation mask for each detected object. Semantic information is encoded using CLIP-based embeddings. While ConceptGraphs explicitly reconstructs per-object 3D geometry, Clio represents objects as spatially grounded semantic entities with associated embeddings and metadata. Clio additionally incorporates a detector confidence score into the object representation [56]. From this point onward, the two methods differ in how object-centric representations are stored, associated, and fused into a persistent semantic map.

ConceptGraphs [17] updates its semantic map by associating newly detected objects with existing ones if the DBSCAN-filtered 3D point cloud exhibits both high semantic similarity, mea-

sured using OpenCLIP [50], and sufficient 3D spatial overlap, quantified via IoU. If these conditions are met, the new point cloud is aligned to the existing object using Iterative Closest Point (ICP) and merged, while semantic embeddings are aggregated over time. Otherwise, a new object node is added to the map. The resulting representation is a 3D scene graph, where nodes correspond to objects and edges encode spatial relationships (e.g., above, next to) derived from relative geometry. Each object is subsequently captioned using a large VLM to generate a human-readable description of its attributes and context [17]. A LLM then processes the set of graph nodes and their captions under a system prompt to refine object descriptions and infer higher-level relationships between objects, by connecting spatial edges between relevant nodes. This enables a range of downstream tasks, including question answering about the scene, robot manipulation, navigation, and localization.

Clio [56] employs a dual-level object-centric memory consisting of a frontend instance buffer and a backend abstract object memory. The frontend maintains a transient buffer of newly detected object instances, each associated with a spatial estimate, a CLIP-based semantic embedding, and a detector confidence score. These instance-level representations are updated at a high frequency and may contain redundant or noisy observations. The backend maintains a compact set of stable, task-relevant object abstractions that serve as long-term semantic memory. In Clio, promotion from instance-level observations to abstract object anchors is governed by a mathematically defined information bottleneck objective operating in semantic embedding space, which compresses redundant observations while preserving task-relevant information. This enables efficient querying by retaining only a small number of semantically meaningful object anchors [57]. During language-guided navigation, semantic reasoning is performed by matching the CLIP embedding of the language query against the abstract object representations, while low-level navigation is handled by a classical planner operating on the geometric occupancy map. While Clio improves efficiency by retaining only task-relevant object abstractions, this task-specific design limits map reusability, as querying non-task objects requires additional map reconstruction. As noted by Jiang et al. [47], DualMap addresses this limitation by adopting a hybrid segmentation strategy for holistic open-vocabulary mapping and by replacing costly inter-object merging with lightweight intra-object consistency checks, enabling persistent semantic memory and more efficient online operation.

Kerr et al. [51] introduced *LERF*, which builds upon Neural Radiance Field (NeRF) [58] to represent a scene as a continuous volumetric function that maps a 3D position $\mathbf{x}$ and viewing direction $\mathbf{d}$ to color and density, $f(\mathbf{x}, \mathbf{d}) \rightarrow (\sigma, \mathbf{c})$, learned from multi-view RGB images with known camera poses. While a standard NeRF reconstructs only geometry and appearance, LERF augments the radiance field with an additional language embedding output, $f(\mathbf{x}, \mathbf{d}) \rightarrow (\sigma, \mathbf{c}, \mathbf{e})$, where $\mathbf{e}$ denotes a language-aligned semantic embedding. This embedding field is learned by distilling image-level semantic features into 3D during training. Specifically, CLIP [4] and DINO [33] embeddings are extracted from each training image at multiple spatial scales and projected into the NeRF via multi-view consistency, allowing each 3D location to store a stable, language-aligned semantic representation. This additional supervision enables dense 3D relevancy maps to be generated at query time by computing similarity between a text embedding and the learned language field. However, LERF requires extensive offline, scene-specific

NeRF training and does not support incremental updates or online exploration, limiting its applicability to real-time or long-term robotic mapping scenarios.

Yamazaki et al. [35] presented *OpenFusion*, a real-time open-vocabulary semantic mapping framework that integrates volumetric Truncated Signed Distance Field (TSDF)-based reconstruction with region-level semantic perception from foundation models. OpenFusion processes incoming RGB images using Segment Everything Everywhere All at Once (SEEM) [53], a promptable VLM capable of zero-shot semantic segmentation based on text or image prompts. For each frame, SEEM [53] produces soft region confidence maps along with a semantic embedding vector for each region.

Depth images and camera poses are fused into a volumetric TSDF map to reconstruct scene geometry. Rather than directly storing semantic embeddings per voxel, OpenFusion associates voxels with lightweight semantic region identifiers with a dictionary mapping each region ID to its corresponding semantic embedding. To establish temporal consistency, the current TSDF map is raycast from the camera pose to render the accumulated semantic regions into the image plane. The rendered regions are then compared with the newly observed regions using geometric overlap and semantic similarity, and region correspondences are solved via a Jonker-Volgenant assignment algorithm [59]. Matched regions are fused by updating confidence scores, while unmatched regions are added as new semantic entries [35].

Semantic embeddings are stored in a global dictionary indexed by region identifiers, rather than per voxel, reducing memory consumption while enabling efficient semantic queries. At query time, a natural-language prompt is embedded using the same VLM, and cosine similarity 7 is computed between the query embedding and the dictionary entries. Regions with the highest similarity scores are retrieved and localized via their associated TSDF geometry. This design enables real-time, open-vocabulary semantic mapping with efficient memory usage. However, OpenFusion does not maintain explicit object instances or support object-centric or instance-level semantic reasoning, which could potentially be addressed by additional voxel clustering strategies. Similar to DualMap [47] and Clio [56], OpenFusion follows a local-global fusion paradigm, where semantic information extracted from individual observations is incrementally associated with a global map representation. However, unlike DualMap and Clio, which explicitly maintain semantic abstractions across time, OpenFusion performs this fusion at the level of region-aligned geometry without constructing persistent object instances or multi-level semantic memory [35, 47, 56].

Alama et al. [15] introduced *RayFronts*, a real-time open-set semantic mapping framework designed to support both fine-grained semantic scene understanding within sensor range and semantic reasoning about regions beyond the depth perception limit. RayFronts represents the environment using a hybrid spatial abstraction consisting of a sparse voxel-based semantic map for observed regions and a set of semantic ray frontiers anchored at map boundaries for unobserved space.

Given posed RGB-D observations, RayFronts first extracts dense, language-aligned visual features using an efficient vision-language encoder based on RADIO [60] with a SIGLIP [61] adapter. Within the sensor range, depth measurements are fused into a sparse voxel grid, where each occupied voxel stores a persistent semantic embedding aggregated over time

via a weighted averaging scheme. This lightweight fusion strategy prioritizes computational efficiency and online operation, in contrast to more complex multi-stage or object-level fusion pipelines [17, 47, 56].

To reason about regions beyond the depth sensing horizon, RayFronts maintains a VDB-based occupancy map that encodes free, occupied, and unknown space. Three-dimensional frontiers are extracted as boundary voxels separating observed and unobserved regions. Instead of associating a single semantic descriptor with each frontier, RayFronts introduces semantic ray frontiers, in which multiple rays are attached to each frontier voxel. Each ray is parameterized by its origin, direction, and a language-aligned visual embedding, capturing semantic evidence observed along that direction in image space.

Semantic rays are discretized using angular bins and incrementally fused over time, allowing multiple distinct semantic hypotheses to coexist at the same frontier without feature collisions. This stands in contrast to VLFM [9], which maintains a single-query, episodic value map conditioned on a specific object prompt. This design enables multi-object and multi-query semantic reasoning in unobserved space and supports rough triangulation of distant semantic entities as exploration progresses. Importantly, both voxel and ray representations store task-agnostic visual embeddings rather than query-specific scores, allowing the semantic map to be queried at arbitrary times using text or image prompts via cosine similarity [15].

RayFronts is explicitly planner-agnostic and does not prescribe a specific exploration strategy. Instead, it provides a persistent semantic scene representation that can be consumed by downstream planners for object search, exploration, or navigation in large-scale and unbounded environments. By decoupling semantic mapping from planning, RayFronts enables flexible integration with a wide range of exploration and decision-making frameworks while maintaining real-time performance in outdoor settings [15].

Schwaiger et al. [54] introduced *OTAS* (Open-vocabulary Token Alignment for Outdoor Segmentation), a training-free semantic segmentation and reconstruction framework designed for unstructured outdoor environments. Unlike prior open-vocabulary mapping approaches that rely on object-centric segmentation priors, OTAS extracts semantic structure directly from intermediate token representations of frozen vision and vision-language foundation models.

OTAS uses DINOv2 [52] to generate dense visual embeddings at the image patch level, which capture visual similarity but are not language-aligned. To obtain language grounding, dense patch-level embeddings are extracted from CLIP [4], which are language-aligned but noisy and view-dependent. The core idea of OTAS is to cluster visually similar patches based on DINOv2 embeddings and align these clusters with CLIP [4] embeddings via masked average pooling, yielding language-grounded semantic regions without relying on object-centric segmentation [54].

Semantic queries are performed by embedding text prompts using CLIP's text encoder and computing cosine similarity against the cluster-level embeddings. Optionally, a frozen mask refinement network such as SAM2 [55] can be used to upsample coarse relevance maps to pixel-level segmentations. For multi-view reconstruction, language-grounded features are projected into 3D using depth and camera poses and fused into a persistent point cloud representation, enabling open-vocabulary querying via cosine similarity. Unlike prior object-centric mapping

approaches such as ConceptGraphs [17] and Clio [56], OTAS does not require explicit object detection or instance segmentation, allowing it to capture amorphous or unstructured semantic entities commonly found in outdoor environments, such as vegetation, terrain types and natural landmarks [54].

2.6 Identified Research Gaps in State of the Art

Table 7 provides a overview of recent methods for open-vocabulary semantic exploration and mapping. The methods are categorized into four main groups: (a) reinforcement learning-based approaches, (b) foundation-model-based exploration, (c) foundation-model-based exploration with persistent semantic mapping, and (d) mapping-centric methods. The second column indicates the type of exploration strategy employed, while the third column highlights whether each method incorporates an explicit exploration-exploitation tradeoff mechanism, i.e., whether the balance between exploring new areas and exploiting known information can be controlled. The fourth and fifth columns denote whether the method supports zero-shot generalization to unseen environments and real-time operation, respectively.The final column indicates whether the method maintains a persistent semantic memory.

The main limitation of reinforcement learning-based approaches lies in their limited adaptability to unseen environments, where structural differences such as room layouts, object appearances, and lighting conditions deviate from the training distribution [11, 26]. These methods typically require extensive retraining or fine-tuning to generalize effectively, which limits their applicability in dynamic or real-world settings. Most importantly, RL-based approaches do not construct persistent semantic memory during exploration, preventing the reuse of acquired knowledge across tasks or environments unless such behavior is explicitly encoded during training [11].

Foundation-model-based exploration methods mitigate this limitation by leveraging pre-trained vision- language models to guide exploration in a zero-shot manner toward semantically relevant regions. VLFM [9] combines classical frontier-based exploration with semantic value maps derived from image-text similarity, enabling efficient zero-shot navigation. However, the semantic representation in VLFM is episodic and reset after each navigation episode, preventing long-term semantic reasoning or refinement over time.

OneMap [16] addresses this limitation by accumulating semantic embeddings in a probabilistic 2D map, enabling persistent semantic querying across exploration episodes. While OneMap integrates semantic similarity maps with open-vocabulary object detectors through consensus filtering, the detector output acts as a hard gating mechanism. Consequently, object hypotheses are only formed when the detector confidence exceeds a predefined threshold, preventing the accumulation of weak but consistent semantic evidence across views when detector confidence remains low.

DualMap [47] improves robustness by introducing explicit object detection and an object-centric dual-map structure that separates short-term observations from long-term semantic anchors. While this design enhances semantic stability, both OneMap [16] and DualMap [47] primarily exploit previously observed semantic information and do not provide mechanisms

Method	Exploration Type	Explicit Exploration-Exploitation Tradeoff	Zero-Shot	Real-Time	Persistent Semantic Memory
(a) Reinforcement learning-based					
ZSON [11]	Reinforcement Learning	✗	✗	✗ (scene specific training)	✗
PONI [25]	Reinforcement Learning	✓	✗	✗ (scene specific training)	✗
(b) Foundation-model-based exploration					
VLFM [9]	Foundation-model	✗	✓	✓	✗
(c) Foundation-model + persistent semantic mapping					
OneMap [16]	Foundation-model	✗	✓	✓	✓
DualMap [47]	Foundation-model	✗	✓	✓	✓
RayFronts [15]	Foundation-model	✗ (planner-agnostic)	✓	✓	✓
(d) Mapping-centric methods					
ConceptGraphs [17]	Mapping-centric	✗	✓	✗	✓
OpenFusion [35]	Mapping-centric	N/A	✓	✓	✓
OTAS [54]	Mapping-centric	N/A	✓	✓	✓
SAGE (this work)	Foundation-model	✓	✓	✓	✓

Table 7: Comparison of open-vocabulary semantic exploration and mapping methods. The table highlights differences in exploration strategy, semantic persistence, zero-shot generalization, and explicit exploration-exploitation control.

to actively balance exploration and exploitation. Semantic reasoning is applied mainly after anchors have been formed, limiting their ability to guide early exploration.

RayFronts [15] extends semantic reasoning beyond observed space by introducing semantic ray frontiers, enabling reasoning about unobserved regions outside the depth sensing range. Although this representation supports persistent semantic querying and multi-object reasoning, the framework is explicitly planner-agnostic and does not define a concrete exploration strategy or an exploration-exploitation control mechanism.

Mapping-centric approaches such as ConceptGraphs [17], OpenFusion [35], and OTAS [54] focus on building rich semantic scene representations but do not address exploration behavior. These methods lack mechanisms for semantic-driven frontier selection or tunable control of exploration based on uncertainty or memory reliability.

In summary, existing approaches either perform semantic exploration without persistent, revisable semantic memory, or construct persistent semantic maps without explicitly guiding exploration. No prior work jointly addresses zero-shot semantic exploration, persistent semantic memory, and an explicit, adjustable exploration-exploitation tradeoff within a unified framework. Furthermore, existing methods rely predominantly on single-source semantic signals, lacking multi-source confidence fusion to robustly suppress false positives during exploration. These identified research gaps motivate the development of **Semantic-Aware Guided Exploration (SAGE)**, which aims to integrate these capabilities into a cohesive system for robust and efficient open-vocabulary semantic exploration.

3 Methods

The method proposed by this thesis combines zero-shot semantic perception with frontier-based exploration and persistent 3D semantic mapping into a unified, closed-loop decision-making framework. Semantic evidence acquired during exploration is continuously fused into a long-term spatial memory, while exploration behavior is adaptively modulated based on the reliability of accumulated semantic beliefs.

An overview of the system architecture is provided in Section 3.1, followed by detailed descriptions of the core components: semantic frontier exploration (Section 3.2), promptable zero-shot detection (Section 3.4), persistent semantic 3D mapping (Section 3.3), the multi-source fusion strategy governing semantic belief updates (Section 3.5), and the behavior tree used to orchestrate semantic-guided exploration and navigation (Section 3.6).

3.1 Architecture Overview

The proposed system follows a modular hybrid architecture that couples semantic perception, geometry-driven exploration, and persistent semantic mapping to enable open-vocabulary object-guided exploration. Figure 1 provides a high-level overview of the components and data flow. The exploration task is specified by a natural language prompt, which defines the semantic target and conditions the detection, exploration, and fusion modules.

Robot observations at time t are represented as RGB-D observations $O_t = \{I_t, D_t\}$, where I_t denotes the RGB image and D_t the depth map. The robot pose in the world frame is denoted by P_t. To reduce computational load, observations are temporally and spatially downsampled prior to further processing.

The pre-processed observations are then fed into three parallel modules. **(a)** The *Memory Module* takes as input O_t and P_t and updates the persistent semantic 3D map M_t^{sem}. **(b)** The *Exploration Module* uses O_t and P_t to generate frontier candidates $F_t = \{f_i\}$ from the current geometric exploration map, denoted as M_t^{exp}, and assigns each candidate a semantic relevance score s_i^{exp} based on vision-language similarity computed with BLIP-2 [27]. **(c)** The *Detection Module* processes O_t to produce promptable zero-shot detections D_t^{det}.

Each module outputs a set of graph nodes representing semantic hypotheses. A graph node is defined as $g_i = (\mathbf{p}_i, s_i)$ with 3D position $\mathbf{p}_i = (x_i, y_i, z_i)$ and an associated semantic score $s_i \in [0, 1]$, where the score is sourced from the respective module, i.e., $s_i \in \{s_i^{\text{mem}}, s_i^{\text{exp}}, s_i^{\text{det}}\}$. Specifically, the memory module produces memory graph nodes G_t^{mem} derived from M_t^{sem}, the exploration module outputs exploration graph nodes G_t^{exp} corresponding to F_t, and the detection module outputs detection graph nodes G_t^{det} obtained from D_t^{det}. This unified graph abstraction enables heterogeneous hypotheses to be compared, filtered, and ranked using a common representation.

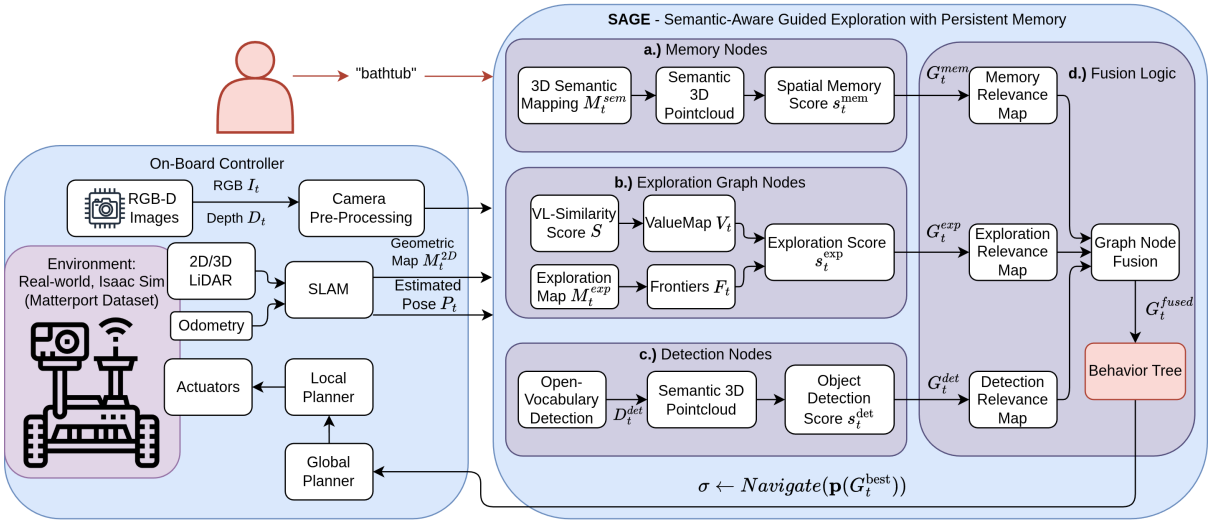


Figure 1: Overview of the **SAGE** pipeline for open-vocabulary semantic exploration. RGB-D observations O_t and poses P_t are processed by three parallel modules: (a) persistent 3D semantic mapping M_t^{sem} (OpenFusion [35]), (b) language-guided frontier scoring over candidates F_t using BLIP-2 [27], and (c) promptable zero-shot detection D_t^{det}. The resulting hypotheses are filtered by relevance maps and fused into graph nodes G_t^{fused}. A behavior tree selects the highest-scoring node G_t^{best} and triggers goal-conditioned navigation.

Prior to fusion, graph nodes are filtered using a relevance map to suppress candidates in already covered regions. The remaining nodes are fused using the multi-source fusion strategy described in Section 3.5, resulting in a unified set of weighted graph nodes G_t^{fused}.

The Behavior Tree described in Section 3.6 selects the next high-level action based on G_t^{fused}. In particular, it selects a goal hypothesis G_t^{best} and invokes a goal-conditioned navigation primitive, $\sigma \leftarrow Navigate(\mathbf{p}(G_t^{\text{best}}))$, where $\sigma \in \{\texttt{success}, \texttt{failure}, \texttt{timeout}\}$ denotes the execution status.

3.2 Semantic Frontier Exploration

Semantic frontier exploration extends classical frontier-based exploration by incorporating semantic relevance derived from vision-language models, enabling task-driven exploration guided by a user-defined semantic prompt. Instead of exploring unknown space uniformly, the robot prioritizes frontiers that are more likely to yield observations relevant to the target concept [2, 9, 15, 62]. Figure 2 illustrates the intermediate map representations and processing stages used to construct the semantic frontier map.

Exploration Occupancy Maps

SAGE maintains three distinct 2D occupancy grids for navigation and exploration: (a) a SLAM map used for localization and navigation [63], (b) an exploration map used exclusively for frontier detection, and (c) an inflated map that suppresses narrow passages and noisy frontier artifacts. This separation decouples stable navigation from task-specific exploration decisions

and prevents semantic exploration logic from modifying the navigation map.

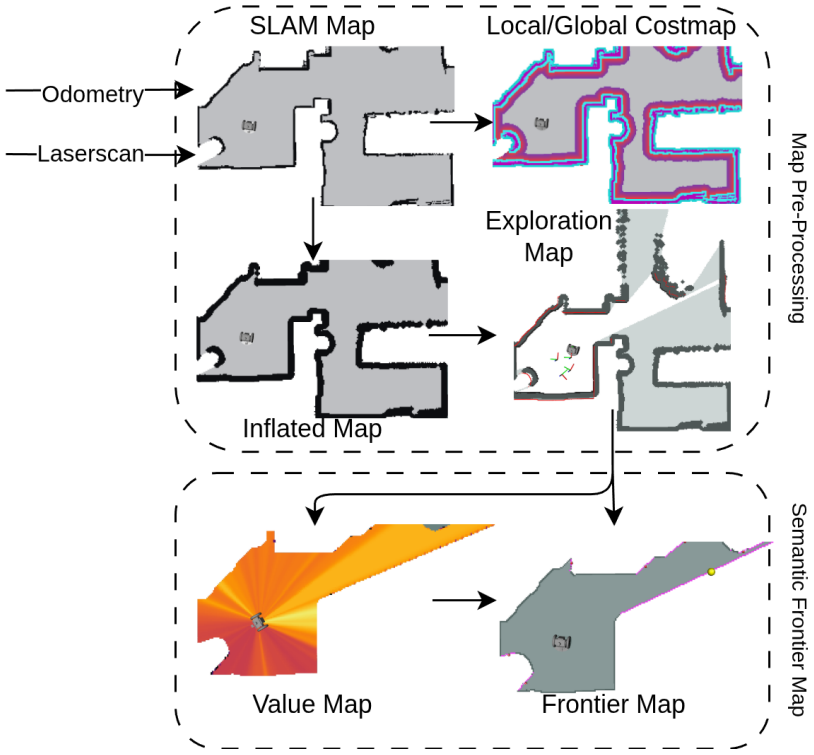


Figure 2: Overview of the map representations used for semantic frontier exploration. The SLAM map is used for localization and navigation, while the exploration map encodes task-specific explored and unexplored regions for frontier detection [63]. An inflated map is used to suppress narrow structures and reduce spurious frontiers. The resulting frontier map is combined with a semantic value map to prioritize exploration toward semantically relevant regions.

Rather than running a separate SLAM instance for exploration, the exploration map is derived directly from the SLAM occupancy grid [63]. Given the robot pose P_t and the SLAM map M_{SLAM}, free space is raycast from the robot into the occupancy grid, using the known sensor model and maximum sensor range, marking traversed cells as explored while preserving unknown regions beyond sensor reach. This raycasting process is applied to all recorded robot poses accumulated during the current task, yielding an exploration map M_{exp} that reflects the explored workspace.

When a new semantic search task is initiated, all stored poses are cleared and the exploration map is rebuilt from scratch, while the SLAM map remains unchanged. This design ensures that exploration decisions are conditioned solely on task-relevant semantic information and prevents bias from previously explored but semantically irrelevant regions.

Frontier Detection and Calculation

Frontiers are defined as the boundary between known free space and unknown regions in the exploration occupancy grid [2] (see Figure 3). This work uses the algorithm outlined in Algorithm 1 to extract and cluster frontiers from the exploration map.

Let $\mathcal{G} \in \{-1, 0, 100\}^{W \times H}$ denote the exploration occupancy grid, where -1 represents un-

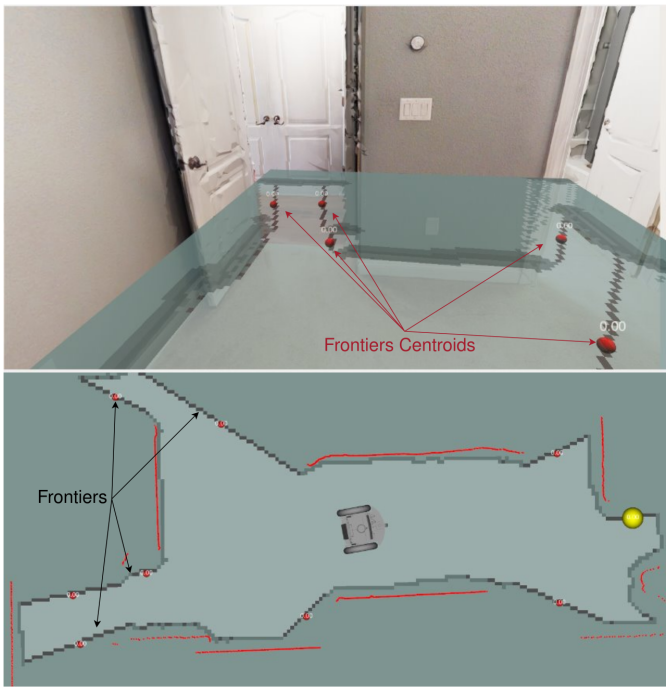


Figure 3: Example of frontier detection on the exploration occupancy grid. Frontiers are identified as free cells adjacent to unknown space and clustered into spatially contiguous regions, with centroids serving as candidate exploration targets.

known space, 0 free space, and 100 occupied space [63]. The set $\mathcal{F}_t$ denotes the set of detected frontier clusters at time step t. Each frontier cluster $\mathcal{P}$ is a set of spatially connected frontier cells. The set $\{\mathbf{c}_i^{t-1}\}$ contains the centroids of frontier clusters detected at the previous time step and is used to maintain temporal consistency via centroid matching.

The extracted frontier cells are clustered using an 8-connected Breadth-First Search (BFS) to group spatially contiguous regions. Clusters that fall within predefined size limits ($N_{\min}$ and $N_{\max}$) are retained, while outliers are discarded. Each valid frontier cluster is represented by its centroid, which serves as the candidate exploration target. Let $\mathcal{P} = \{\mathbf{p}_j \in \mathbb{R}^2 \mid j = 1, \ldots, |\mathcal{P}|\}$ denote the set of grid cell coordinates belonging to a frontier cluster. The centroid $\mathbf{c} \in \mathbb{R}^2$ is defined as the arithmetic mean of the spatial coordinates of all cells belonging to the frontier cluster. Frontier identity matching is performed using nearest-neighbor association in Euclidean space, where a frontier centroid is assigned the ID of the closest previously observed centroid within distance $d_{\text{match}} \in \mathbb{R}$, otherwise a new frontier ID is created.

$$\mathbf{c} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} \mathbf{p} \tag{10}$$

Equation 10 yields a single representative location that approximates the geometric center of the frontier region. This centroid is used as the navigation target for frontier-based exploration and as the reference position for semantic scoring.

The frontier centroids are tracked over time by matching them to previously detected frontiers based on spatial proximity, allowing for consistent identification of persistent frontiers across time steps. To extract the semantically most relevant frontiers, each frontier is scored using the

Algorithm 1 Geometric frontier extraction and clustering from the exploration occupancy grid

function EXTRACTFRONTIERS($\mathcal{G}$, $N_{\min}$, $N_{\max}$, d_{match}, $\{\mathbf{c}_i^{t-1}\}$)
  // Initialize frontier set
  $\mathcal{F}_t \leftarrow \emptyset$
  for all cells (x, y) with $\mathcal{G}(x, y) = 0$ **do**
    if any 4-neighbor of (x, y) is unknown **then**
      Mark (x, y) as frontier
  for all unvisited frontier cells (x, y) **do**
    // 8-connected frontier clustering
    Grow a cluster $\mathcal{P}$ via BFS
    if $N_{\min} \leq |\mathcal{P}| \leq N_{\max}$ **then**
      // Cluster centroid
      $\mathbf{c} \leftarrow \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} \mathbf{p}$
      // Temporal frontier association
      Match $\mathbf{c}$ to $\{\mathbf{c}_i^{t-1}\}$ within $d_{\text{match}} \in \mathbb{R}$
      if no match found **then**
        // Frontier initialization
        Assign new frontier ID
      // Valid frontier update
      Add $(\mathcal{P}, \mathbf{c})$ to $\mathcal{F}_t$
  return $\mathcal{F}_t$

value map generated by the VLM as described below [9]. The scoring procedure is outlined in Algorithm 2 and Algorithm 3. Let $\mathcal{C} = \{(x_i, y_i, s_i)\}$ denote the semantic value map represented as a set of 2D cells with associated semantic scores s_i, obtained by temporally aggregating cosine similarity values between vision-language model embeddings and the user-defined text prompt (Section 3.2). For a value map cell $q \in \mathcal{C}$, $s(q)$ denotes the semantic similarity score stored at that cell.

The value map is a 2D grid in which each cell stores temporally aggregated cosine similarity scores (see Section 3.2) between VLM embeddings of scene observations and a user-defined text prompt. Let $\mathcal{V} = \{(x_i, y_i, s_i)\}$ denote the value map, where (x_i, y_i) are grid coordinates and $s_i \in \mathbb{R}$ is the associated semantic similarity score.

To score a frontier, its centroid $\mathbf{c} \in \mathbb{R}^2$ is projected into the value map coordinate frame. The semantic score of the frontier is obtained by querying the maximum similarity score within a fixed-radius neighborhood around $\mathbf{c}$, as described in Algorithm 2. If no valid score exists within this neighborhood, indicating that the region has not yet been observed, the frontier is marked as unobserved. Using the maximum response emphasizes strong semantic evidence and reduces sensitivity to low-confidence activations under partial observations, consistent with prior semantic frontier scoring approaches [9].

Algorithm 3 summarizes the construction of semantic frontier graph nodes by combining geometric frontier extraction with semantic scoring.

Algorithm 2 Value map query for frontier scoring

function GETSCOREFROMVALUEMAP($\mathcal{C}$, $\mathbf{p}$)
  // Local semantic support radius
  $r \leftarrow 0.3$
  $s_{\max} \leftarrow 0$
  observed $\leftarrow$ false
  for all points $q \in \mathcal{C}$ **do**
    if $\|q - \mathbf{p}\|_2 < r$ **then**
      // Max-pooling of semantic scores
      $s_{\max} \leftarrow \max(s_{\max}, s(q))$
      observed $\leftarrow$ true
  if observed **then**
    // Frontier supported by semantic observations
    return $(\text{observed} = \text{true},\ \text{score} = s_{\max})$
  else
    // Unobserved frontier region
    return $(\text{observed} = \text{false})$

Algorithm 3 Construction of semantic frontier graph nodes

function UPDATESEMANTICFRONTIERS($\mathcal{G}$, $\mathcal{V}$, $\{\mathbf{c}_i^{t-1}\}$)
  // Geometric frontier extraction
  $\mathcal{F}_t \leftarrow \text{EXTRACTFRONTIERS}(\mathcal{G}, N_{\min}, N_{\max}, d_{\text{match}}, \{\mathbf{c}_i^{t-1}\})$
  for all frontier $f \in \mathcal{F}_t$ **do**
    // Representative frontier position
    $\mathbf{c} \leftarrow \text{centroid}(f)$
    // Semantic value lookup
    $(\text{observed}, s) \leftarrow \text{GETSCOREFROMVALUEMAP}(\mathcal{V}, \mathbf{c})$
    Create graph node n
    n.id $\leftarrow f$.id
    n.position $\leftarrow \mathbf{c}$
    n.score $\leftarrow s$
    n.observed $\leftarrow$ observed
    // Semantic frontier node
    Add n to graph
  // For downstream task and visualization
  Publish frontier graph

The final step involves creating graph nodes for each frontier, encapsulating their ID, position, semantic score, and observation status. These semantic frontier graph nodes form the primary input to the fusion strategy and behavior tree described in Sections 3.5 and 3.6.

Value Map Generation using Vision-Language Models

The value map can be interpreted as an analogy to gradient ascent in deep reinforcement learning, where the robot seeks to maximize an expected semantic reward by navigating toward regions with high relevance to a target concept [9, 11, 14, 43]. In this work, the value map represents the slope of a semantic reward function. In contrast to classical gradient ascent, movement toward regions of high semantic relevance is constrained by obstacles and unknown space. Consequently, geometrically derived frontiers serve as feasible navigation targets that guide the robot toward high-value regions while ensuring safe traversal, thereby preventing convergence to unreachable local maxima [14]. Figure 4 illustrates this analogy, showing the semantic reward landscape and the role of frontier-based navigation (see Chapter 3.5 for the reward definition).

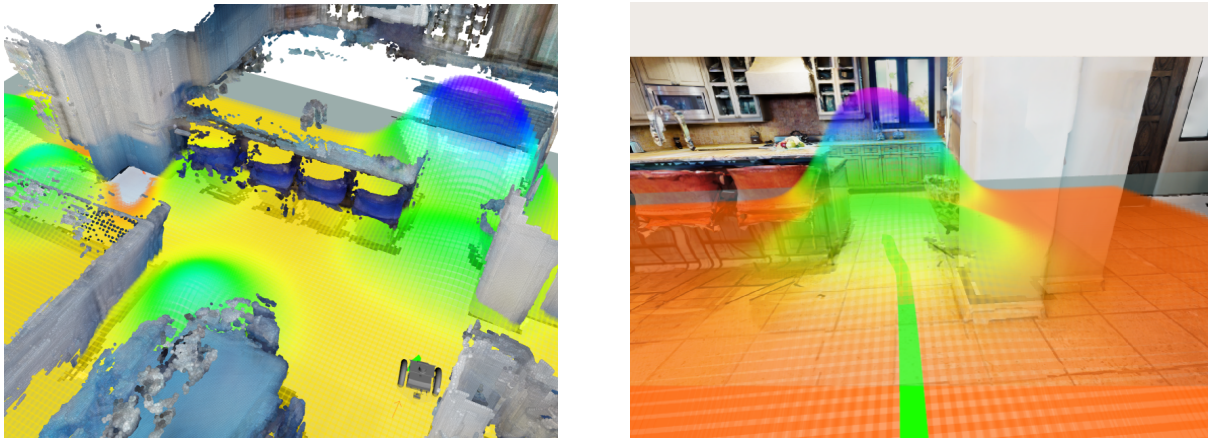


Figure 4: Conceptual visualization of the gradient-ascent analogy for semantic frontier exploration. Semantic relevance is modeled as a folded reward surface over spatial coordinates (x, y), where frontier and memory graph nodes are elevated according to their semantic scores (illustrated for the search of an "`oven in a kitchen`"). Geometric frontiers and obstacles constrain feasible ascent directions, guiding exploration toward semantically promising and navigable regions.

The value map is generated using a pre-trained vision-language model, specifically BLIP-2 [27], which computes semantic similarity between visual observations and a user-defined text prompt. The value map node subscribes to RGB images, robot poses, and a global occupancy grid produced by a LiDAR-based SLAM system. This differs from prior semantic frontier approaches such as VLFM [9], which rely on depth camera projections and odometry-based local maps. By leveraging LiDAR-based SLAM, the proposed system maintains a globally consistent exploration map with reduced drift, enabling persistent semantic value accumulation over long trajectories. Figure 5 illustrates the overall value map generation pipeline.

Upon receiving an RGB image and a text prompt, the BLIP-2 service computes an image embedding and a text embedding using its pre-trained visual and textual encoders. The input image is divided into a grid of fixed-size patches, which are flattened, added with positional embeddings, and linearly projected before being processed by a vision transformer to capture spatial and semantic context [27, 64]. Similarly, the text prompt is tokenized into subword

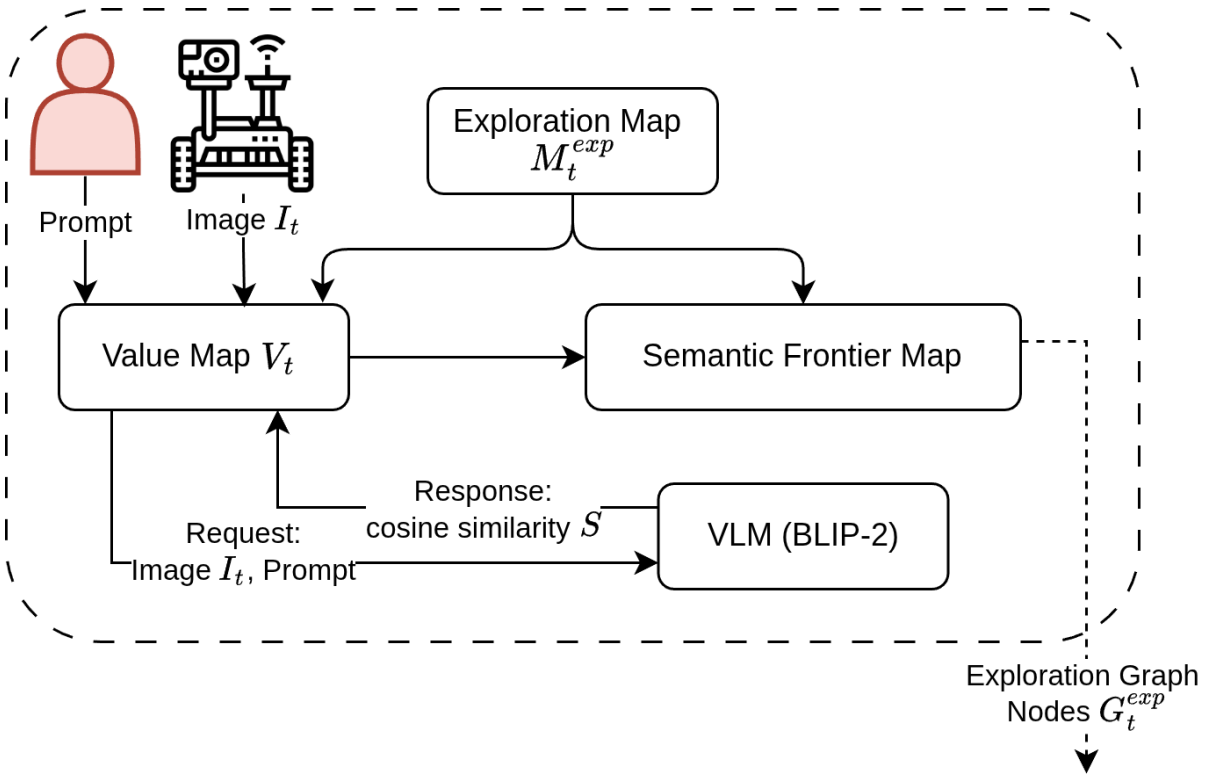


Figure 5: Value map generation pipeline using the BLIP-2 vision-language model for computing image-text cosine similarity.

units, embedded, and processed by a language transformer to model semantic and syntactic relationships.

Therefore, both embeddings are projected using learned projection matrices W_I and W_T into a common embedding space (see Equation 11):

$$E_I = W_I f_I(I), \qquad E_T = W_T f_T(T) \tag{11}$$

where $f_I(\cdot)$ and $f_T(\cdot)$ denote the visual and textual encoders, respectively. The projection matrices W_I and W_T are learned during pre-training using an Image–Text Contrastive (ITC) objective, which optimizes the cosine similarity between matching image-text pairs while pushing apart non-matching pairs [4, 27]. This training procedure aligns visual and textual representations in a shared semantic embedding space, enabling direct similarity comparison via cosine similarity. The projected embeddings are subsequently L2-normalized to unit length, which is required for cosine similarity computation.

The cosine similarity score S between the normalized image embedding $\hat{E}_I$ and text embedding $\hat{E}_T$ is computed as:

$$S = \hat{E}_I \cdot \hat{E}_T = \frac{E_I}{\|E_I\|_2} \cdot \frac{E_T}{\|E_T\|_2} = \cos(\hat{E}_I, \hat{E}_T) \tag{12}$$

Figure 6 illustrates this image-text similarity computation, where visual observations are embedded by the BLIP-2 image encoder and compared against a user-defined text prompt in a shared semantic embedding space. Although cosine similarity is theoretically bounded in

the interval $[-1, 1]$, in practice ITC-based VLMs yield similarity scores concentrated in a narrow positive range [4, 27]. This continuous score serves as a measure of semantic relevance between the current visual observation and the target concept.

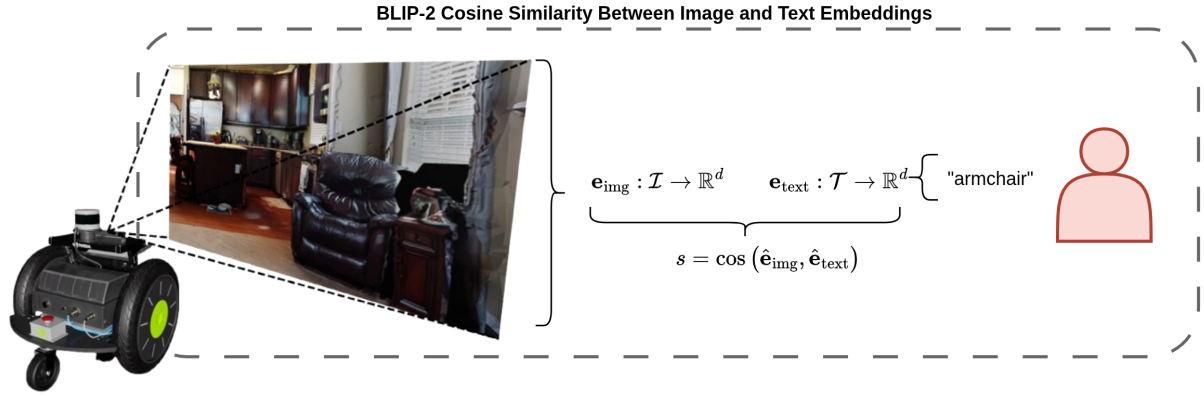


Figure 6: Image-text cosine similarity computation using the BLIP-2 ITC head. An RGB observation is embedded by the visual encoder and compared against a user-defined text prompt in a shared semantic embedding space [27].

Cosine Similarity to Value Map Projection

The computed cosine similarity score is integrated into a persistent 2D semantic value map in a pose- and visibility-aware manner. Let V_{t-1} denote the semantic value map and C_{t-1} the associated confidence map at time step $t-1$, and let s_t be the cosine similarity score obtained from the VLM at the current robot pose $\mathbf{p}_t$.

The update procedure, summarized in Algorithm 4, consists of three conceptual stages: (a) motion-gated temporal decay, (b) visibility-aware observation selection, and (c) confidence-weighted fusion. The following update procedure closely follows the semantic value map formulation proposed in VLFM [9].

(a) Temporal decay is applied to both the value map and the confidence map only if the robot has translated more than a predefined threshold δ_{move} since the previous update. This motion-gated decay prevents repeated observations from dominating the map when the robot remains stationary or performs pure rotations, while still allowing outdated semantic evidence to fade over time when the robot explores new regions. The decay factor $\lambda \in [0, 1]$ controls the persistence of past observations and prevents oscillation between multiple similar frontiers by gradually reducing the influence of stale semantic evidence when the robot revisits similar viewpoints [9].

(b) The set of map cells that can be updated at time step t is determined by computing a top-down visibility mask $M_{\text{fov}}(\mathbf{p}_t)$ using raytracing within the current field of view, instead of relying solely on depth camera projections based on odometry [9, 21]. Only cells that are geometrically visible from the robot's pose and not occluded by obstacles are considered for update. For these visible cells, an instantaneous observation confidence map $C_{\text{obs}}(\mathbf{p}_t, M_{\text{fov}})$ is computed, which models the reliability of the current observation as a function of the sensor geometry. Cells near the center of the FOV are assigned higher confidence, while confidence

Algorithm 4 2D Value Map Update using Vision-Language Model Similarity Scores

function UPDATESEMANTICVALUEMAP(V_{t-1}, C_{t-1}, s_t, $\mathbf{p}_t$)
  // **(a) Motion-gated temporal decay**
  if $\|\mathbf{p}_t - \mathbf{p}_{t-1}\|_2 > \delta_{\text{move}}$ **then**
    $V_{t-1} \leftarrow \lambda V_{t-1}$
    $C_{t-1} \leftarrow \lambda C_{t-1}$
  // **(b) Visibility and observation confidence**
  Compute visibility mask $M_{\text{fov}}(\mathbf{p}_t)$ via raytracing
  Compute confidence map $C_{\text{obs}}(\mathbf{p}_t, M_{\text{fov}})$
  // **(c) Confidence-weighted fusion**
  for all cells (i, j) with $M_{\text{fov}}(i, j) = 1$ **do**
    $v \leftarrow V_{t-1}(i, j)$
    $c \leftarrow C_{t-1}(i, j)$
    $c_{\text{new}} \leftarrow C_{\text{obs}}(i, j)$
    if $c + c_{\text{new}} = 0$ **then**
      continue
    $V_t(i, j) \leftarrow v + \alpha\, c_{\text{new}}(s_t - v)$
    $C_t(i, j) \leftarrow \max(\lambda c,\ c_{\text{new}})$
  return $V_t,\ C_t$

decreases toward the periphery due to reduced resolution and increased distortion. This behavior is modeled using a Gaussian weighting function over the angular deviation from the camera's principal viewing direction. The observation confidence assigned to a visible cell (i, j) is computed as

$$C_{\text{obs}}(i, j) = \mathrm{e}^{-\frac{1}{2}\left(\frac{\Delta\theta(i,j)}{\sigma}\right)^2}, \tag{13}$$

where $\Delta\theta(i, j)$ denotes the angular difference between the viewing ray toward cell (i, j) and the robot's forward-facing direction, and σ controls the sharpness of the confidence decay within the FOV. Smaller values of σ result in a narrower high-confidence region centered around the optical axis, while larger values produce a more uniform confidence distribution. This confidence formulation follows the angular weighting strategy used in VLFM to model observation reliability across the FOV [9].

(c) For each visible cell (i, j), the semantic value stored in the map is updated toward the current similarity score s_t using a confidence-weighted fusion rule. Let $V_{t-1}(i, j)$ and $C_{t-1}(i, j)$ denote the semantic value and confidence stored at cell (i, j) before the update, and let $C_{\text{obs}}(i, j)$ denote the instantaneous observation confidence computed from the current FOV.

The semantic value update is formulated as a weighted interpolation between the previous value and the current similarity score (see Equation (14)):

$$V_t(i, j) = V_{t-1}(i, j) + \alpha\, C_{\text{obs}}(i, j)\big(s_t - V_{t-1}(i, j)\big), \tag{14}$$

where $\alpha \in [0, 1]$ is an update gain that controls how strongly new observations influence the

existing value map. Higher observation confidence leads to a stronger correction toward the current similarity score, while low-confidence observations have only a minor effect. In parallel, the confidence map is updated to preserve strong observations over time. Specifically, the confidence assigned to each cell is defined in Equation (15):

$$C_t(i,j) = \max\big(\lambda\, C_{t-1}(i,j),\ C_{\text{obs}}(i,j)\big), \tag{15}$$

where $\lambda \in [0,1]$ is the temporal decay factor applied when the robot has translated since the previous update. Using a max operation ensures that regions which have been observed with high confidence remain influential even after decay, preventing repeated low-confidence observations from diluting reliable semantic evidence [9].

Together, Equations (14) and (15) implement a persistent, confidence-aware fusion mechanism that incrementally integrates semantic similarity scores into a spatially consistent value map.

Figure 7 illustrates a generated value map for the prompts “Bed”, “TV”, and the zero-shot prompt “A door leading to a bed”, demonstrating how prompt design can bias exploration toward multiple targets or semantically useful transition regions.

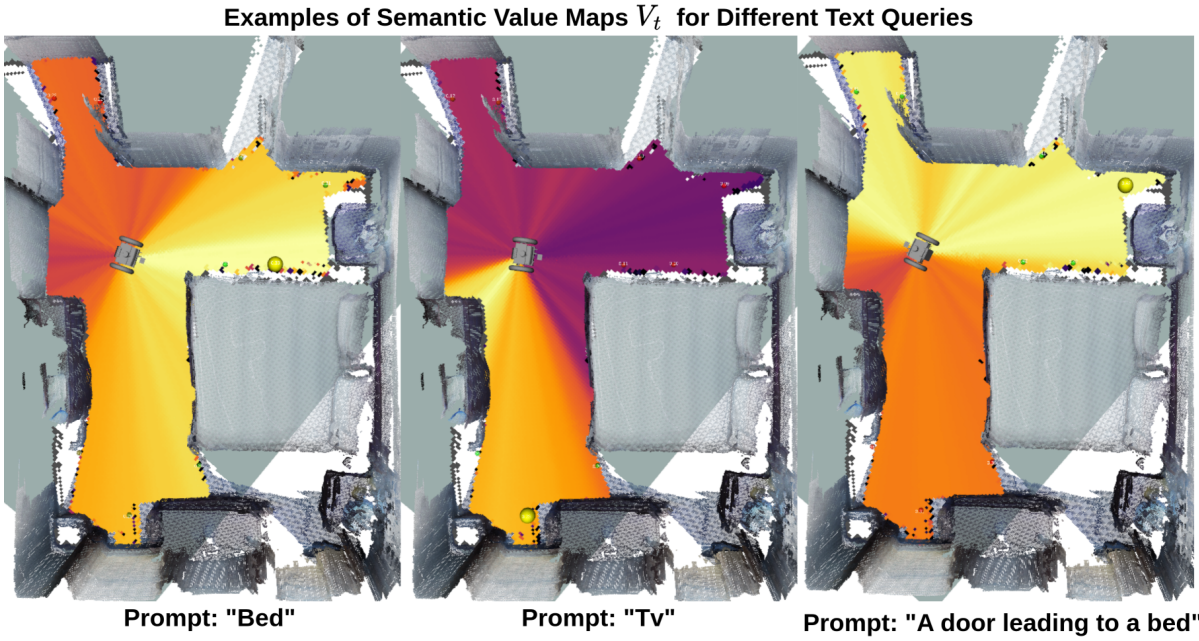


Figure 7: Example value maps generated using BLIP-2 for different text prompts. The value maps highlight regions of high semantic relevance to the prompts “Bed”, “TV”, and “A door leading to a bed”.

These scored frontiers are then encapsulated as graph nodes and passed to the fusion strategy (Section 3.5) for integration with memory and detection modules.

3.3 Persistent Semantic 3D Mapping

The semantic frontier centroids described in Section 3.2 offer short-term exploration targets based on immediate observations. However, to enable effective multi-object search over multiple objects, the system requires a persistent memory that retains information about previously observed objects and their spatial locations throughout the exploration task [16, 17, 47]. The proposed system incorporates a semantic 3D mapping module that constructs a global, persistent semantic representation from RGB-D observations and produces object-level hypotheses in the form of graph nodes. These memory graph nodes are subsequently integrated into the fusion strategy (see Section 3.5) to inform exploration and detection decisions.

At the time of implementation, OpenFusion [35] represented a suitable open-source framework for real-time, open-vocabulary semantic 3D mapping. Its balance between reconstruction fidelity and computational efficiency makes it suitable for onboard deployment on mobile robotic platforms [35]. As discussed in the state-of-the-art analysis (Chapter 2.5), OpenFusion enables zero-shot semantic mapping but does not provide an object-centric abstraction. This work extends OpenFusion with object-level semantic clustering and graph node generation, yielding a persistent semantic memory tailored to multi-object search and long-horizon exploration tasks.

Global Map Construction

OpenFusion [35] was originally designed for offline semantic reconstruction from pre-recorded RGB-D sequences, such as ScanNet [65] and Replica [40]. In contrast, this work adapts OpenFusion for online operation on a mobile robot by integrating it with a LiDAR-based SLAM system that provides globally consistent pose estimates (see Figure 8).

The wrapper subscribes to RGB images I_t, depth images D_t, camera intrinsics K, and SLAM-based robot poses P_t. To limit computational overhead and avoid redundant observations, RGB-D (I_t, D_t) frames are forwarded to OpenFusion only if the robot pose P_t differs sufficiently from previously integrated viewpoints. This pose filtering reduces redundant TSDF updates from near-identical viewpoints, mitigates over-integration artifacts, and ensures efficient use of the fixed voxel budget. Specifically, an input tuple $\{(I_t, D_t, P_t)\}$ is accepted only if the robot has translated or rotated more than a predefined threshold δ_{move} and represents a novel viewpoint relative to the existing map.

Upon initialization, OpenFusion is configured using the current camera intrinsics and a fixed voxel resolution and voxel budget, which together define the available memory for semantic mapping [35]. For each accepted RGB-D observation (I_t, D_t, P_t), the data are fused into a global TSDF volume using the SLAM-based pose estimate, yielding a globally consistent 3D reconstruction with reduced drift compared to odometry-only approaches [21].

Each point in the semantic point cloud is associated with a cosine similarity score between the user-defined text prompt and the corresponding visual observation. The visual embeddings are obtained using SEEM [53], a pre-trained VLM optimized for dense, pixel-level representations. SEEM produces a high-dimensional embedding for each pixel in the RGB image, which is then compared to the text prompt embedding using cosine similarity. By projecting these pixel-wise similarity scores into 3D space via the aligned RGB-D observations, a semantic

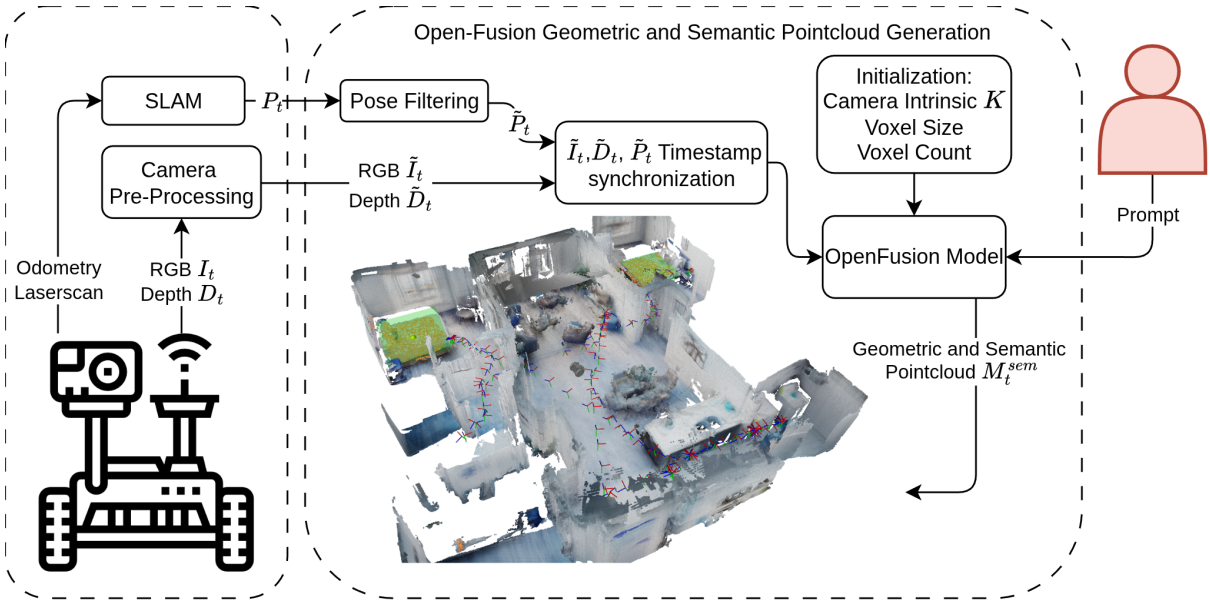


Figure 8: OpenFusion-based pipeline for persistent semantic 3D mapping. RGB images I_t, depth D_t images, and LiDAR-based SLAM poses P_t are synchronized and filtered before integration into a global TSDF volume. On-demand semantic queries generate semantic and panoptic point clouds, which are further processed to extract object-level hypotheses for the fusion strategy.

point cloud is obtained in which each point encodes its semantic relevance with respect to the query.

In practice, only the most semantically relevant predictions are retained by selecting a subset of high-confidence regions, controlled by a top-k selection criterion. This selection acts as a filtering mechanism rather than a strict object count, limiting the number of candidate regions considered for further processing.

Based on these high-confidence regions, SEEM performs instance-aware grouping, yielding a panoptic point cloud in which each 3D point is assigned both a semantic label and an instance identifier [53]. This panoptic representation allows spatially distinct object instances of the same semantic class to be differentiated. While OpenFusion provides the underlying 3D reconstruction and voxel management, semantic embeddings are computed externally using SEEM and fused into the 3D map via aligned RGB-D projections [35, 53].

These instance-level clusters are subsequently aggregated into object-centric memory graph nodes, which are consumed by the fusion strategy for long-horizon semantic exploration and object search.

Semantic Clustering and Graph Node Generation

The semantic point cloud produced by OpenFusion encodes dense geometric structure together with a per-point semantic relevance score derived from vision-language similarity [35, 53]. To extract object-centric hypotheses suitable for long-horizon exploration and persistent memory reasoning, a dedicated clustering and aggregation stage is applied [17].

Prior to clustering, the semantic point cloud is spatially downsampled using a voxel grid filter

with a fixed leaf size. This operation reduces sensor noise and computational complexity while preserving the coarse geometry of potential object instances [66, 67].

Let the semantic point cloud be defined as $\mathcal{P} = \{(\mathbf{x}_i, s_i)\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^3$ denotes the 3D position of point i in the global map frame, $s_i \in [0, 1]$ its associated semantic similarity score, and N the total number of points. The objective of clustering is to partition $\mathcal{P}$ into a set of spatially coherent clusters

$$\mathcal{C} = \{C_1, \ldots, C_M\}, \quad C_k \subset \mathcal{P}, \tag{16}$$

where each cluster C_k corresponds to a single object hypothesis and M denotes the number of extracted clusters. Clustering is performed using Euclidean distance in 3D space [21, 67]. Two points $\mathbf{x}_i$ and $\mathbf{x}_j$ are assigned to the same cluster if $\|\mathbf{x}_i - \mathbf{x}_j\|_2 \leq \varepsilon_{\text{cluster}}$, where $\varepsilon_{\text{cluster}} > 0$ is a fixed spatial distance threshold controlling the maximum cluster extent. To suppress spurious noise clusters and overly large, diffuse regions, only clusters satisfying $N_{\min} \leq |C_k| \leq N_{\max}$ are retained, where $|C_k|$ denotes the number of points in cluster C_k and $N_{\min}, N_{\max}$ are user-defined bounds.

Each retained cluster C_k is represented geometrically by its centroid (see Equation 17), which serves as the spatial location of the corresponding object hypothesis.

$$\mathbf{c}_k = \frac{1}{|C_k|} \sum_{(\mathbf{x}_i, s_i) \in C_k} \mathbf{x}_i, \tag{17}$$

To assign a semantic confidence score to each cluster, the per-point similarity scores must be aggregated into a single representative value. Although the input point clouds already exclude background geometry due to semantic filtering by OpenFusion [35] and SEEM [53], the distribution of similarity scores within a semantic cluster remains highly non-uniform. Even within a single object instance, confidence varies significantly due to partial observations, occlusions, viewing angle effects, depth noise, and heterogeneous visual evidence across object surfaces. As a result, only a subset of points typically exhibits strong semantic alignment with the query, while the remaining points carry weaker but still relevant signals.

Therefore, a naïve aggregation using the arithmetic mean is insufficient for this setting. Within a semantic cluster, the mean is dominated by weakly informative points and systematically underestimates objects that are only partially visible or observed from suboptimal viewpoints. To obtain a estimate of semantic relevance, a percentile-based aggregation is employed instead [68]. Let $\mathcal{S}_k = \{s_i \mid (\mathbf{x}_i, s_i) \in C_k\}$ denote the set of similarity scores within cluster C_k. The cluster-level semantic confidence is defined as the 75-th percentile of this set (see Equation 18):

$$\tilde{s}_k = \text{percentile}_p(\mathcal{S}_k), \tag{18}$$

Selecting the 75th percentile emphasizes clusters for which a substantial fraction of points exhibits consistently high semantic similarity, while reducing sensitivity to isolated spurious activations. Lower percentiles (e.g., the median) are overly conservative in the presence of localized but strong semantic evidence, whereas higher percentiles (e.g., the maximum) are more sensitive to noise [68].

To further favor spatially consistent object hypotheses without allowing large clusters to dominate purely by size, the percentile score is modulated by a logarithmic cluster size factor (see Equation 19):

$$s_k = \tilde{s}_k \cdot \log(|C_k| + 1), \tag{19}$$

where the logarithmic term grows sublinearly with cluster size. In the semantic-only setting considered here, this factor reflects the spatial support of the semantic hypothesis rather than compensating for background clutter, thereby favoring compact objects supported by a coherent extent of evidence.

Finally, each cluster C_k is mapped to a memory graph node

$$n_k = (\mathbf{c}_k, s_k), \tag{20}$$

where $\mathbf{c}_k$ denotes the spatial location of the object hypothesis and s_k its aggregated semantic relevance score.

3.4 Promptable Zero-Shot Detection

To enable goal-directed object search without prior knowledge of object categories, the exploration system integrates a promptable open-vocabulary object detection model. Such models leverage large-scale vision-language pre-training to localize objects specified by arbitrary user-defined prompts, enabling zero-shot generalization beyond a fixed training vocabulary [4].

Table 8 compares representative state-of-the-art detection and segmentation models with respect to their zero-shot capability, supported prompt modalities, output representations, foundation architectures, and suitability for real-time robotic operation.

For integration into a 3D semantic fusion pipeline, 2D detections must be lifted into 3D space using depth measurements and camera intrinsics. A common approach consists of projecting instance-segmented pixels into 3D and clustering the resulting point sets to obtain object hypotheses [70, 71]. Consequently, the detection model must provide instance-level segmentation masks, confidence scores, and zero-shot generalization.

Object detection approaches can be categorized into three classes. (a) Closed-vocabulary methods rely on a fixed set of object classes defined during training and cannot generalize to unseen categories, as exemplified by classical detectors such as Mask R-CNN [30] or YOLOv7 [72]. (b) Open-vocabulary methods extend detection to arbitrary categories specified via text prompts, enabling zero-shot recognition but typically lacking fine-grained instance segmentation [28, 32, 69]. (c) Promptable zero-shot models further generalize this paradigm by supporting multiple prompt modalities, such as textual descriptions and visual reference images, allowing context-aware and flexible object specification [42, 53].

Promptable segmentation models such as SAM [29] and SEEM [53] provide powerful instance mask generation capabilities from visual or textual prompts. However, these approaches rely on heavy transformer-based architectures with prompt-conditioned decoding,

Method	Zero-Shot	Prompt Type	Instance Segmentation	Bounding Boxes	Foundation Model	Real-Time
Grounding Dino [28]	✓	Text	✗	✓	DINO + BERT	✗
SAM [29]	✓	Visual	✓	✗	ViT	✗
SEEM [53]	✓	Text, Visual, Spatial	✓	✗	ViT + Text Encoder	✗
GLIP [32]	✓	Text	✗	✓	Swin Transformer + BERT	✗
OWL-ViT [69]	✓	Text	✗	✓	ViT + CLIP	✗
Mask R-CNN [30]	✗	None	✓	✓	ResNet + FPN	✗
YOLO-E [42]	✓	Text, Image	✓	✓	YOLO + CLIP [4]	✓

Table 8: Comparison of object detection and segmentation models with respect to zero-shot capability, prompt modality, output representation, and real-time suitability. Real-time capability is defined as achieving at least 10 Hz inference on a single GPU, which is sufficient for low-dynamic robotic exploration tasks.

leading to inference latency that prevents bounded real-time execution within robotic control loops.

Related to the proposed object search system, VLFM [9] addresses zero-shot object search by integrating multiple perception modules rather than introducing a single unified detection model. Specifically, VLFM combines a closed-vocabulary detector for fast and reliable detections, an open-vocabulary grounding model that operates at a lower frequency to enable zero-shot detection, and a separate segmentation network to extract object centroids used for navigation toward the target. While this modular design facilitates flexible semantic reasoning, it requires multiple sequential inference stages, resulting in substantial computational overhead and limiting its suitability for real-time deployment.

In contrast, YOLO-E [42] unifies promptable zero-shot detection and instance segmentation within a single, efficient architecture. Building upon the YOLO framework [72] and incorporating CLIP-based vision-language alignment, YOLO-E achieves competitive detection accuracy while maintaining real-time inference. On the LVIS benchmark [73], YOLO-E reports average precision values of 35.2 and 33.7 for text and image prompts, respectively, surpassing GLIPv2 and GroundingDINO, which achieve 29.0 and 27.4 AP. Moreover, the `YOLOE-11-L` variant employs only 26 M parameters for text prompting and 32 M parameters for image prompting, compared to over 170 M parameters for GroundingDINO and more than 230 M parameters for

GLIPv2. With reported inference speeds of 130 Hz on an NVIDIA T4 GPU and 39.2 Hz on mobile phone hardware, YOLO-E satisfies the bounded-latency requirements of online robotic exploration.

These properties make YOLO-E particularly well suited for semantic exploration scenarios that require flexible object specification, instance-level spatial reasoning, and real-time operation under limited computational resources.

Open-Vocabulary Object Detection with YOLO-E

In this work, YOLO-E [42] is used as a promptable zero-shot object detection module within the exploration system. Incoming Red Green Blue–Depth (RGB-D) observations are preprocessed by throttling the input frame rate and resizing images to the required network resolution in order to reduce computational load while maintaining sufficient perceptual fidelity.

Given a user-defined text prompt and the current RGB image, YOLO-E [42] performs a single forward pass to produce a set of 2D bounding boxes, confidence scores, and corresponding instance segmentation masks for all objects matching the prompt (see Figure 9). For visualization purposes, an overlay image containing bounding boxes, class labels, and confidence scores is generated. In addition, an instance ID mask is published to differentiate between multiple detected objects of the same category, and a per-pixel score mask is produced to assign semantic confidence values to segmented image regions.

Using the scored 2D segmentation mask, the instance ID mask, the aligned depth image, and the camera intrinsic parameters, detected object pixels are projected into 3D space (see Section 3.4). This projection yields a semantic point cloud in which each point $\mathbf{p}_i = (x_i, y_i, z_i)$ is associated with a semantic confidence value and an instance identifier. A subsequent clustering stage aggregates the semantic point cloud into object-level hypotheses by grouping spatially coherent points belonging to the same instance. For each cluster, a 3D centroid and a mean confidence score are computed.

The resulting 3D object hypotheses are encapsulated as detection graph nodes and forwarded to the fusion strategy for integration with the persistent semantic memory (see Section 3.5).

Depth-Based 3D Localization

Let a pixel (u, v) in the image plane be represented in homogeneous coordinates as $\tilde{\mathbf{p}} = [u, v, 1]^\top$, where u and v denote the horizontal and vertical pixel indices, respectively. Equation (21) defines the camera intrinsic matrix $\mathbf{K} \in \mathbb{R}^{3\times3}$, parameterized by the focal lengths f_x, f_y and the principal point coordinates c_x, c_y [74]. These parameters are obtained by calibrating the RGB-D camera prior to deployment with standard calibration techniques [75].

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{21}$$

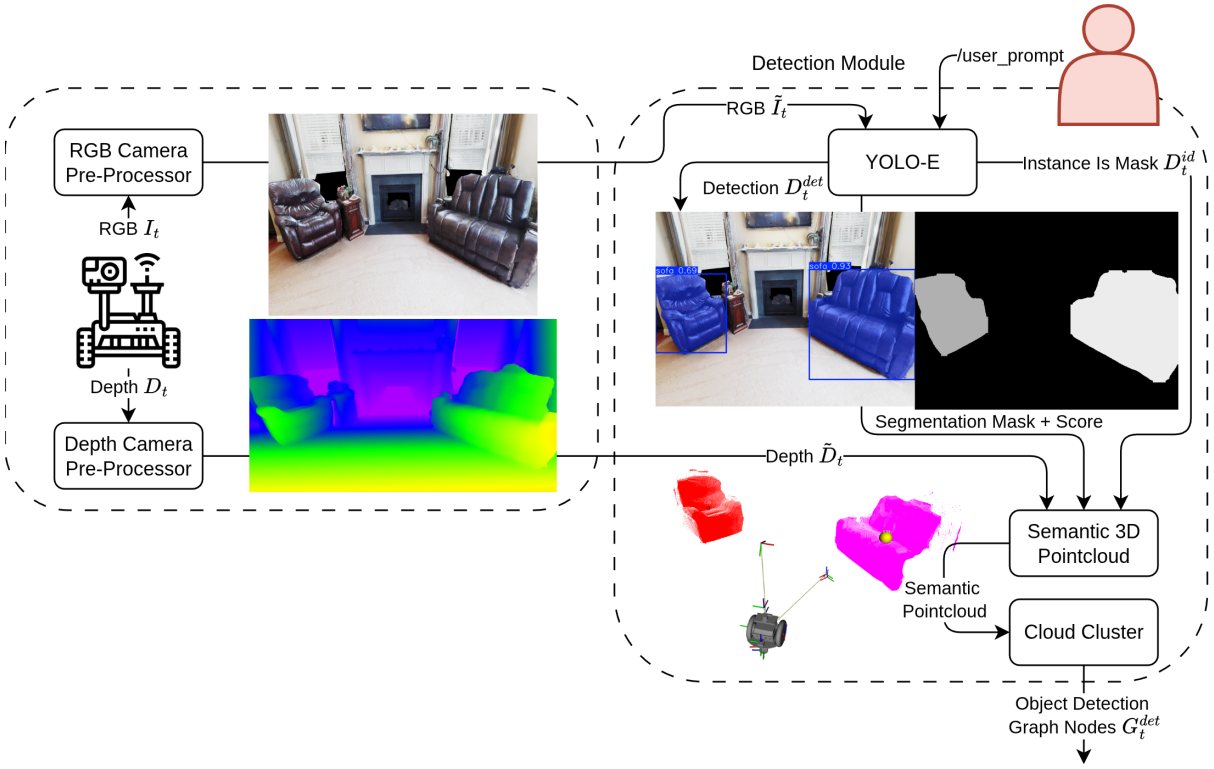


Figure 9: Promptable open-vocabulary object detection and 3D hypothesis generation pipeline using YOLO-E [42].

Given a metric depth value $z \in \mathbb{R}^+$ at pixel (u, v), the corresponding 3D point in the camera coordinate frame is obtained via Equation (22), which is illustrated in Figure 10 [74].

$$\mathbf{P}_{\text{cam}} = z\,\mathbf{K}^{-1}\tilde{\mathbf{p}} \quad = z \begin{bmatrix} \frac{1}{f_x} & 0 & -\frac{c_x}{f_x} \\ 0 & \frac{1}{f_y} & -\frac{c_y}{f_y} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{(u-c_x)}{f_x} z \\ \frac{(v-c_y)}{f_y} z \\ z \end{bmatrix} \tag{22}$$

To express the 3D point in the global map frame, the homogeneous transformation $\mathbf{T}_{\text{map}\leftarrow\text{cam}} \in SE(3)$ is applied (see Equation (23)).

$$\mathbf{T}_{\text{map}\leftarrow\text{cam}} = \begin{bmatrix} \mathbf{R}_{\text{map}\leftarrow\text{cam}} & \mathbf{t}_{\text{map}\leftarrow\text{cam}} \\ \mathbf{0}^\top & 1 \end{bmatrix} \tag{23}$$

where $\mathbf{R} \in SO(3)$ denotes the rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ the translation vector. The transformation $\mathbf{T}_{\text{map}\leftarrow\text{cam}}$ is obtained from the robot pose estimate provided by the localization and mapping system.

Using homogeneous coordinates $\tilde{\mathbf{P}}_{\text{cam}} = [\mathbf{P}_{\text{cam}}^\top, 1]^\top$, the 3D point in the global map frame is obtained in Equation (24).

$$\tilde{\mathbf{P}}_{\text{map}} = \mathbf{T}_{\text{map}\leftarrow\text{cam}} \tilde{\mathbf{P}}_{\text{cam}}. \tag{24}$$

Dropping the homogeneous component yields the final 3D point $\mathbf{P}_{\text{map}} \in \mathbb{R}^3$ expressed in the global map coordinate system.

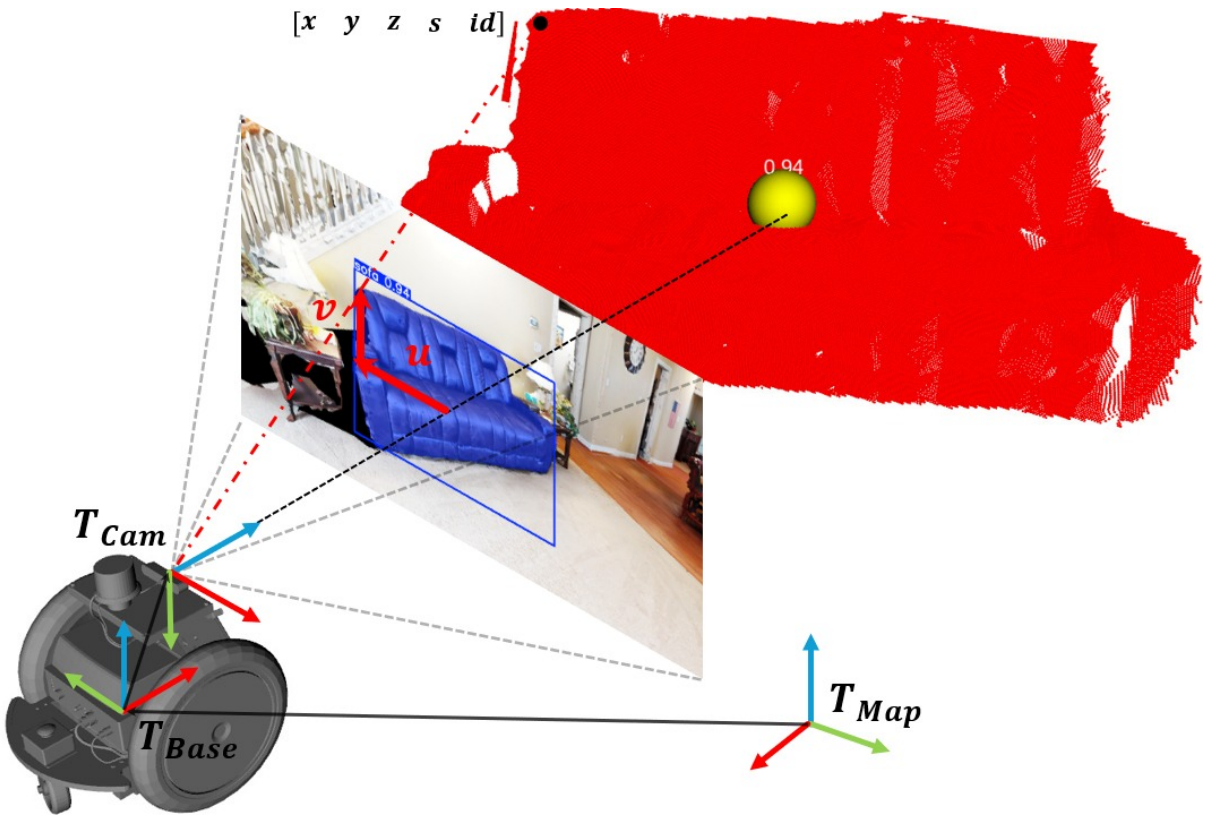


Figure 10: Back-projection of a 2D YOLO-E detection into 3D space using the pinhole camera model. Image-space coordinates are transformed from the camera frame to the global map frame, yielding a 3D semantic graph node with an associated confidence score.

Each pixel (u, v) is additionally associated with a semantic confidence score $s(u, v) \in [0, 1]$ and an instance identifier $\mathrm{id}(u, v) \in \mathbb{N}$, obtained from the instance segmentation output of the visual detector.

While the geometric back-projection and coordinate transformation affect only the spatial components of the point, the semantic attributes are propagated unchanged from the image domain to the 3D representation. Consequently, each 3D point in the global map frame is represented as an augmented semantic point. Each pixel (u, v) is associated with a semantic confidence score and an instance identifier produced by the detector at the current camera pose (see Equation (25)).

$$\mathbf{q}_{\mathrm{map}} = \begin{bmatrix} \mathbf{P}_{\mathrm{map}} \\ s \\ \mathrm{id} \end{bmatrix} = \begin{bmatrix} x_{\mathrm{map}} \\ y_{\mathrm{map}} \\ z_{\mathrm{map}} \\ s(u, v) \\ \mathrm{id}(u, v) \end{bmatrix} \tag{25}$$

This formulation preserves object-level information across the 2D-3D lifting process, enabling instance-consistent aggregation, clustering, and persistent semantic reasoning directly in the global map frame.

Similar to the semantic clustering process described in Section 3.3, the augmented 3D points

are grouped into object-level clusters based on spatial proximity and shared instance identifiers. Each cluster is then summarized as a detection graph node, which encapsulates the 3D centroid and aggregated confidence score of the detected object hypothesis. However, instead of using the percentile-based aggregation employed for memory nodes, the mean confidence score is computed for detection nodes, reflecting the typically higher reliability of promptable detection outputs [42, 72] compared to open-vocabulary semantic mapping [35].

3.5 Semantic Fusion Strategy

Analogous to the gradient ascent formulation introduced in Section 3.2, the fusion strategy defines the similarity score function that guides the robot's exploration and exploitation behavior. Its purpose is to combine heterogeneous sources of semantic information into a unified decision space that prioritizes promising regions for object search while maintaining navigational feasibility.

The fusion strategy integrates three complementary types of graph nodes: (a) exploration frontier nodes, representing geometrically reachable but semantically unexplored regions, (b) persistent memory nodes, encoding accumulated semantic evidence from past observations, and (c) detection graph nodes, derived from promptable zero-shot object detections. By jointly reasoning over these sources, the system balances exploration of unknown space with exploitation of semantically relevant hypotheses.

Prior to fusion, graph nodes from all sources are filtered using a relevance map (Section 3.5). The relevance map is updated continuously during exploration and suppresses graph nodes located in regions that have already been observed and deemed irrelevant to the current semantic query. This mechanism prevents oscillatory behavior and repeated visitation of low-value areas.

After relevance filtering, the remaining graph nodes are fused in two stages, as illustrated in Figure 11. First, exploration frontier nodes and memory nodes are combined using an exploration-memory weighting scheme (Section 3.5), which explicitly trades off short-term exploration incentives against long-term semantic beliefs. Second, detection graph nodes are fused with semantic evidence from the value map and persistent memory via a multi-source detection fusion mechanism (Section 3.5), yielding object hypotheses with aggregated confidence estimates.

The fusion process produces two sets of graph nodes: an exploration-memory node set, which guides exploratory navigation, and a detection node set, which represents high-confidence object hypotheses. Both sets are forwarded to the behavior-tree-based navigation policy (Section 3.6), which orchestrates low-level path planning and robot control to navigate toward high-reward regions identified by the fusion strategy.

Exploration-Memory Weighting

A graph node is defined as $n_i = (\mathbf{x}_i, s_i)$, where $\mathbf{x}_i \in \mathbb{R}^3$ denotes the spatial position of the node in the global map frame and $s_i \in [0, 1]$ represents its associated semantic relevance

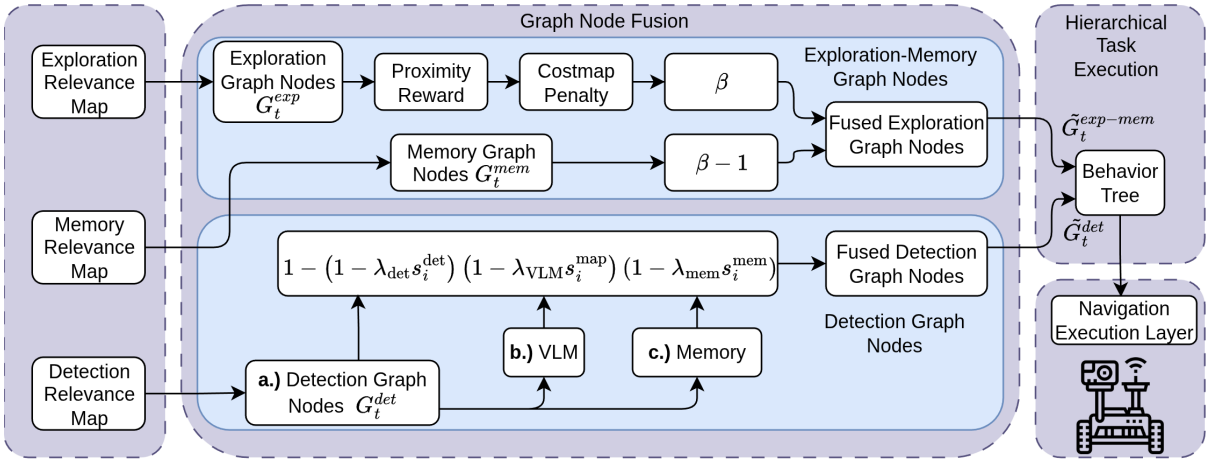


Figure 11: Overview of the fusion pipeline, illustrating how exploration frontiers, persistent memory nodes, and promptable detection hypotheses are filtered, weighted, and fused into a unified graph representation used for navigation decisions.

score. For the purpose of exploration-memory weighting, only the planar component $\mathbf{x}_i = (x_i, y_i) \in \mathbb{R}^2$ is considered, since navigation and motion planning are performed on a 2D planar representation. Height variations are handled implicitly by the mapping and localization system and are therefore omitted at this stage.

Following classical frontier-based exploration strategies [2, 62], graph nodes that are spatially closer to the robot are favored to encourage efficient and stable exploration behavior. To this end, the Euclidean distance between a graph node n_i and the current robot position $\mathbf{x}_r = (x_r, y_r) \in \mathbb{R}^2$ is computed in Equation (26).

$$d_i = \|\mathbf{x}_i - \mathbf{x}_r\|_2, \tag{26}$$

This distance is used to define a proximity persistence weight $w_{\text{prox}}(n_i)$, which increases the influence of nearby nodes while smoothly decaying with distance. The weighting function is formulated as a Gaussian kernel in Equation (27):

$$w_{\text{prox}}(n_i) = 1 + \alpha_{\text{prox}} \exp\left(-\frac{1}{2}\left(\frac{d_i}{r_{\text{prox}}}\right)^2\right), \tag{27}$$

where $\alpha_{\text{prox}} > 0$ controls the maximum amplification of nearby nodes and r_{prox} defines the spatial decay radius of the Gaussian kernel. This formulation softly biases the exploration policy toward frontiers close to the robot while preserving the relative ordering induced by the semantic score s_i.

In addition to proximity, navigation feasibility is taken into account by penalizing graph nodes located in regions with high traversal cost. Similar to the approach proposed in [13], a local neighborhood around each node is defined in Equation (28):

$$\Omega_i = \{\mathbf{x}_j \mid \|\mathbf{x}_j - \mathbf{x}_i\|_2 \leq r_{\text{cost}}\}, \tag{28}$$

where r_{cost} denotes the radius within which costmap values are aggregated and $\mathbf{x}_j$ corresponds to individual costmap cell locations [76]. The locally aggregated cost $\bar{c}_i$ is computed as

a Gaussian-weighted average in Equation (29):

$$\bar{c}_i = \frac{\sum\limits_{j \in \Omega_i} c_j \, \exp\!\left(-\frac{\|\mathbf{x}_j - \mathbf{x}_i\|_2^2}{2\sigma_c^2}\right)}{\sum\limits_{j \in \Omega_i} \exp\!\left(-\frac{\|\mathbf{x}_j - \mathbf{x}_i\|_2^2}{2\sigma_c^2}\right)}, \tag{29}$$

where c_j denotes the raw costmap value at location $\mathbf{x}_j$ and σ_c controls the spatial smoothing of the aggregation. The Gaussian-weighted average is used, because it emphasizes costmap values closer to the node location while still incorporating information from the broader neighborhood. This aggregation mirrors the spatial smoothing and inflation mechanisms commonly used in navigation costmaps [76].

To ensure numerical stability and comparability, the aggregated cost is normalized to the interval $[0, 1]$. The normalized cost is then transformed into a penalty weight in Equation (30):

$$w_{\text{cost}}(n_i) = \max(0.3,\ 1 - \lambda_{\text{cost}} \hat{c}_i)\,, \tag{30}$$

where λ_{cost} determines the influence of the costmap on the final score. A minimum weight of 0.3 is enforced to avoid complete suppression of nodes in high-cost regions, ensuring that candidate hypotheses remain selectable even when traversability estimates are conservative.

After defining the proximity and cost-based weighting terms, exploration frontier nodes are reweighted according to Equation (31):

$$s_i^{\text{explore}} = s_i \cdot \beta \cdot w_{\text{prox}}(n_i) \cdot w_{\text{cost}}(n_i), \tag{31}$$

where $\beta \in [0, 1]$ controls the trade-off between exploration and memory exploitation. Such explicit exploration-exploitation trade-offs are widely used in frontier-based and potential-based exploration methods [25]. Larger values of β prioritize the discovery of new areas, while smaller values emphasize revisiting previously observed high-relevance regions. In all experiments, β is treated as a global hyperparameter that can be tuned to adjust the exploration-exploitation balance. Conversely, memory graph nodes are downscaled during exploration in Equation (32).

$$s_i^{\text{memory}} = (1 - \beta)\, s_i, \tag{32}$$

Multi-Source Detection Fusion

The aim of multi-source detection fusion is to estimate the confidence of detection graph nodes by combining evidence from three complementary sources: (a) the raw confidence score provided by the promptable detection vision model, (b) the semantic value map generated from the cosine similarity of the VLM, and (c) the persistent memory graph encoding accumulated semantic beliefs (see Figure 11). A detection hypothesis must be present for fusion to occur: value map and memory cues can only reinforce an existing detection and cannot generate object hypotheses on their own. This design treats the value map and memory terms as supportive priors rather than hypothesis generators, because dense similarity fields and accumulated semantic beliefs lack instance-level segmentation and are therefore not sufficiently specific

to propose discrete object hypotheses without inducing a large number of ambiguous candidates [9, 14]. This prevents false positives caused by dense semantic similarity alone while still allowing consistent contextual and memory-based evidence to increase confidence. In order to effectively combine these heterogeneous sources, a weighted Noisy-OR model is employed. It is well suited for fusing independent probabilistic evidence [77] (see Equation (33)).

$$s_i^{\text{fused}} = 1 - \prod_{k \in \{\text{det, map, mem}\}} \left(1 - \lambda_k \, s_i^{(k)}\right) \tag{33}$$

where $s_i^{(k)}$ denotes the confidence score from source k and $\lambda_k \in [0, 1]$ represents the corresponding source weight. The weights λ_k allow tuning the influence of each source on the final fused confidence score. In this work, the weights are treated as fixed hyperparameters and tuned empirically. Equation (34) shows the explicit form of the weighted Noisy-OR fusion for the three considered sources: detection confidence, value map score, and memory relevance.

$$s_i^{\text{fused}} = 1 - \left(1 - \lambda_{\text{det}} s_i^{\text{det}}\right) \left(1 - \lambda_{\text{map}} s_i^{\text{map}}\right) \left(1 - \lambda_{\text{mem}} s_i^{\text{mem}}\right) \tag{34}$$

The value map score s_i^{map} is computed by aggregating the semantic similarity values from the value map within a local neighborhood around the detection node (see Equation (35) and Equation (36)):

$$\mathcal{V}_i = \{\mathbf{v}_j \mid \|\mathbf{v}_j - \mathbf{x}_i\|_2 < r_v\} \tag{35}$$

r_v defines the radius of the neighborhood around the detection node location $\mathbf{x}_i$, and $\mathbf{v}_j$ denotes individual value map cell locations. The value map score is then computed as the mean similarity within this neighborhood in Equation (36):

$$s_i^{\text{map}} = \frac{1}{|\mathcal{V}_i|} \sum_{\mathbf{v}_j \in \mathcal{V}_i} v_j \tag{36}$$

where v_j denotes the semantic similarity value at location $\mathbf{v}_j$ and $|\mathcal{V}_i|$ is the cardinality of the neighborhood set. Intuitively, if a memory node is near to the detection node, then the detection is more likely to be correct, as the memory node encodes prior evidence for the object being present in that location. Consequently, the memory relevance score s_i^{mem} is computed by aggregating the confidence scores of nearby memory nodes using a Gaussian kernel in Equation (37):

$$s_i^{\text{mem}} = \frac{\sum_{m_k \in \mathcal{N}_{\text{memory}}} s_k \, e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_k\|_2^2}{2\sigma_m^2}}}{\sum_{m_k \in \mathcal{N}_{\text{memory}}} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_k\|_2^2}{2\sigma_m^2}}} \tag{37}$$

where $\mathcal{N}_{\text{memory}}$ denotes the set of memory graph nodes, $\mathbf{x}_k$ and s_k are the spatial position and confidence score of memory node m_k, and σ_m controls the spatial decay of influence. The Gaussian kernel ensures that nearby memory evidence contributes more strongly than distant or outdated observations. The memory relevance term acts as a soft spatial prior and does not perform explicit belief updating.

The final graph node representation after multi-source detection fusion is given in Equation (38):

$$n_i = \left(\mathbf{x}_i,\ s_i^{\text{det}},\ s_i^{\text{map}},\ s_i^{\text{mem}},\ s_i^{\text{fused}}\right) \tag{38}$$

where $\mathbf{x}_i$ denotes the spatial position of the detection node, s_i^{det} is the raw detection confidence from the vision model, s_i^{map} is the aggregated value map score, s_i^{mem} is the aggregated memory relevance score, and s_i^{fused} is the final fused confidence score after multi-source detection fusion.

Instead of relying solely on the raw detection confidence from a single vision model [9], or solely on thresholding cosine similarity or memory evidence [14], the proposed multi-source detection fusion approach combines complementary evidence from multiple independent sources. Under the Noisy-Or formulation, aggregating multiple signals increases the probability of selecting a hypothesis when any source provides strong evidence, which is expected to improve sensitivity to true positives and reduce missed detections. At the same time, this aggregation can increase the acceptance rate of ambiguous hypotheses, potentially lowering precision due to additional false positives. Consequently, the fusion acts as a tunable mechanism to trade precision for recall via the source weights and decision thresholds [77].

Relevance Filtering and Node Suppression

To prevent repeated visitation of low-value areas and oscillatory behavior, a relevance map is maintained during exploration to suppress graph nodes located in regions that have already been observed and deemed irrelevant to the current semantic query. The relevance map is constructed analogously to the value map described in Section 3.2, but instead of accumulating cosine similarity scores, it encodes spatial visibility and observation history.

While the semantic value map relies exclusively on an angular confidence weighting to model the camera FOV (see Equation (13)), the relevance map extends this formulation by additionally incorporating a radial weighting term. This allows the relevance map to suppress regions that have already been observed within both the angular FOV and a bounded spatial range.

The radial weighting component is defined in Equation (39):

$$w_{\text{rad}}(\mathbf{x}) = e^{-\frac{\|\mathbf{x}-\mathbf{x}_r\|_2^2}{2\sigma_r^2}}, \tag{39}$$

where $\mathbf{x}_r$ denotes the robot position and σ_r controls the spatial decay of relevance with increasing distance. The angular weighting term is defined in Equation (13) in Section 3.2.

The combined relevance weighting is obtained as the product of both components, resulting in high relevance values for regions that are close to the robot and aligned with the camera's optical axis.

Three instances of the relevance map are maintained with different parameterizations for exploration, detection, and memory graph nodes, respectively. Graph nodes whose relevance value exceeds a predefined threshold are suppressed, indicating that the corresponding region has already been sufficiently observed.

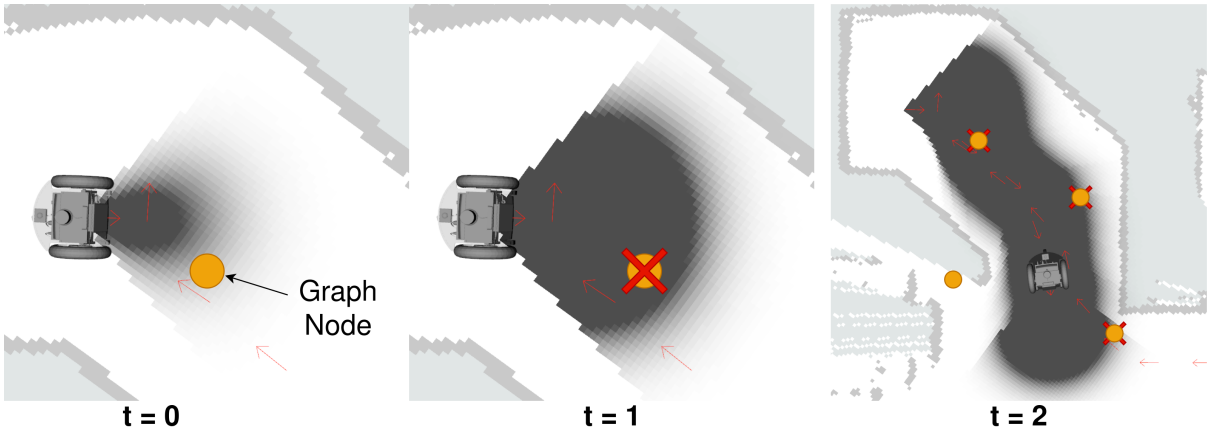


Figure 12: Illustration of relevance-based graph node suppression over time. At $t{=}0$, a graph node is generated within the robot's current field of view. As the robot observes the region ($t{=}1$), the relevance map accumulates visibility evidence, causing the node to be suppressed once the area is deemed sufficiently observed. At $t{=}2$, only graph nodes outside previously observed regions or associated with high-confidence detections remain active, preventing oscillatory exploration while preserving strong semantic hypotheses.

To reduce missed detections, graph nodes whose detection confidence exceeds a high-confidence threshold (80% in all experiments) are exempt from relevance-based suppression and are directly republished. This mechanism enables the robot to revisit locations with strong object evidence while avoiding repeated exploration of low-value regions. Figure 12 illustrates the relevance filtering process over three consecutive timestamps.

3.6 Behavior Tree for Semantic-Guided Exploration

A Behavior Tree (BT) [78] is employed to manage the high-level task structure for semantic-guided exploration. The BT orchestrates the robot's actions based on the fused graph nodes generated by the fusion strategy (Section 3.5), integrating initialization, detection, exploration, and termination behaviors into a unified decision-making framework. Low-level motion execution and collision avoidance are delegated to layered costmap planning [79], allowing the BT to focus exclusively on task sequencing and semantic reasoning.

High-Level Task Structure

At startup, the BT initializes the system by publishing the user-defined object query and clearing previously accumulated semantic and relevance maps. The BT then enters a detection branch, where it continuously monitors the fused detection graph nodes. If an object hypothesis exceeds a predefined confidence threshold for a sustained duration of one second, the robot navigates to the corresponding graph node, realigns its pose for optimal viewpoint alignment, and performs a verification step before capturing an image of the object.

If no object is detected, the BT transitions into the exploration branch. In this branch, the robot observes all available exploration-memory graph nodes, consisting of unexplored fron-

tiers and persistent memory nodes, by performing in-place rotations to gather additional observations. Following this observation phase, the robot navigates towards the highest-valued exploration-memory node as determined by the fusion strategy. This closed-loop process repeats until either a target object is detected or a predefined time limit is reached.

An overview of this decision logic is illustrated as a flowchart in Figure 13, while the full BT structure is provided in Figure 26 in the Appendix.

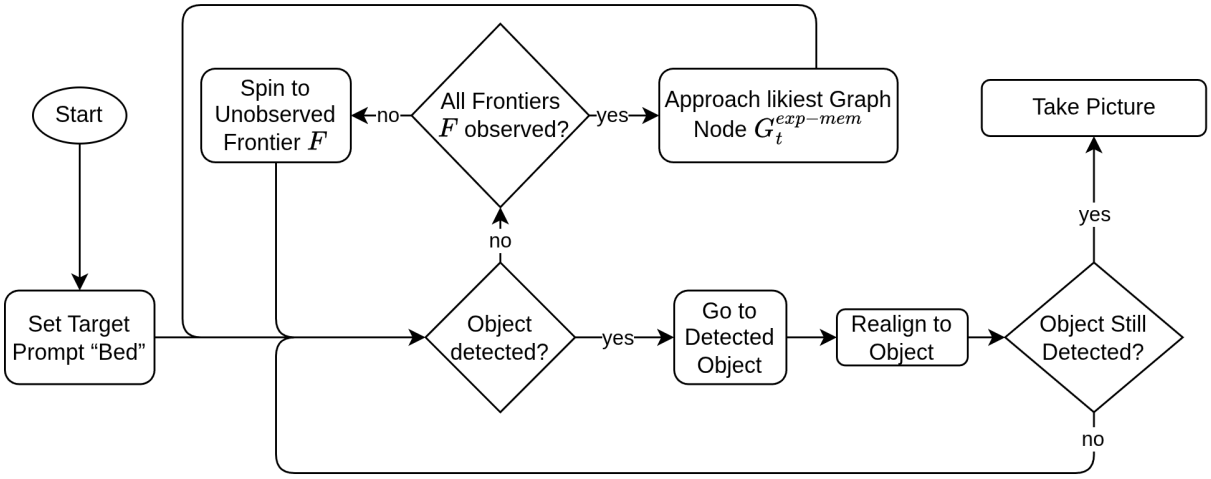


Figure 13: High-level behavior tree flowchart for BT-based semantic-guided exploration. The robot alternates between detection and exploration, navigating to detected objects when confident and otherwise exploring unobserved frontiers until the search terminates.

Integration with the Navigation Stack

The multi-object search behavior implemented by the BT is encapsulated within a high-level action server, enabling a user or external agent to initiate a search task with a specified object prompt. Optional parameters include a zero-shot prompt for value map generation, a time limit, and soft and hard confidence thresholds for object verification.

The BT interfaces with the navigation stack to execute motion-related actions. Specifically, the `GoToGraphNode` behavior uses the `/navigate_to_pose` action interface to issue navigation goals, while the `ObserveGraphNodes` and `RealignToObject` behavior nodes utilize the `/spin` action interface to perform in-place rotations for environmental observation and viewpoint refinement. The navigation stack is responsible for path planning, obstacle avoidance, and low-level control, thereby decoupling motion execution from high-level decision-making [76, 78].

In order to for the robot to approach target objects in cluettered environments, an `ApproachPoseAdjustor` decorator node computes an alternative approach pose based on the target graph node location, the robot's current pose, and the global costmap. If no valid approach pose can be found, the original graph node position is used as the navigation goal. For further details on the Behavior Tree structure and implementation, please refer to Appendix A.

4 Implementation Detail

In this chapter, the implementation of the used methods is evaluated using simulations. First, the chosen simulation environment is described in Section 4.1. Next, the dataset used for training and evaluation is presented in Section 4.2. Section 4.3 outlines the software frameworks and hardware components employed in this work. Finally, the evaluation pipeline and metrics used to assess SAGE are detailed in Sections 4.4 and 4.5, respectively.

4.1 Simulation Environment

HabitatSim [41] has been widely adopted in embodied Artificial Intelligence (AI) research due to its efficient rendering pipeline and support for large-scale datasets such as Matterport3D and HM3D [46]. However, for this work, NVIDIA Isaac Simulation Platform (IsaacSim) was selected as the primary simulation platform because it provides high-fidelity photorealistic rendering, accurate physics simulation, and realistic sensor models. These properties are essential for reducing the sim-to-real gap and for ensuring that the proposed semantic exploration and fusion strategies transfer reliably to real robotic systems [80].

All experiments are conducted in IsaacSim version 5.0.0 by NVIDIA [80], leveraging its GPU-accelerated rendering, physically based simulation, and tight integration with ROS 2 [81]. The pre-installed `Carter` mobile robot is used as the navigation platform and is equipped with an RGB-D camera, a 2D LiDAR sensor, and wheel odometry. The robot is controlled through ROS 2 nodes communicating with IsaacSim via Fast DDS [82], enabling direct reuse of the navigation and perception stack employed in real-world deployment.

IsaacSim further supports importing complex indoor environments derived from Matterport3D [46], which are used as evaluation scenes in this work. To accommodate the physical footprint of the mobile robot and to ensure stable navigation behavior, all Matterport3D scenes are uniformly scaled by 50% in each spatial dimension to accommodate the physical footprint of the mobile robot and ensure stable navigation behavior.

Alternative simulators such as HabitatSim [41], MuJoCo [83], and Ignition Gazebo [84] were considered. However, these platforms either lack full physics-based mobile robot simulation, photorealistic sensor rendering, or seamless ROS 2 integration, which are required by the proposed system.

4.2 Dataset

Prior work such as VLFM [9], CoW [14], and OneMap [16] has demonstrated the effectiveness of Matterport3D-derived indoor environments for semantic exploration and object-goal

navigation tasks. This experimental protocol closely follows the Habitat Navigation Challenge 2023 [85], which evaluates semantic navigation methods on the HM3D dataset [86].

Accordingly, this work employs Matterport3D-based indoor scenes [46] as the primary source of evaluation environments. The dataset provides a diverse set of real-world indoor spaces with high-quality 3D reconstructions and semantic annotations, making it well suited for evaluating open-vocabulary semantic exploration methods.

Due to the custom simulation setup in Isaac Sim, all baseline methods are re-implemented to ensure a fair comparison under consistent conditions, following the evaluation protocol of the Habitat Navigation Challenge 2023 (see Section 4.4) [85]. All evaluations are conducted on the HM3Dv2 validation split, which contains scenes with diverse layouts, object distributions, and levels of complexity.

The following scenes from the Matterport3D dataset are used in this work:

- `00800-TEEsavR23oF`
- `00813-svBbv1Pavdk`
- `00814-p53SfW6mjZe`
- `00824-Dd4bFSTQ8gi`
- `00848-ziup5kvtCCR`
- `00876-mv2HUxq3B53`

These scenes are also used by Yokoyama et al. [9] and Busch et al. [16], enabling direct and reproducible comparison of experimental results.

4.3 Used Software and Hardware

The proposed semantic exploration and fusion framework is implemented using Robot Operating System 2 (ROS2) (Humble Hawksbill) as the middleware for inter-node communication and system integration. ROS2 provides a modular, distributed architecture that allows perception, mapping, fusion, and exploration components to operate as independent nodes [87].

For navigation, localization, and obstacle avoidance, the Navigation 2 (Nav2) stack is employed [76]. In combination with Simultaneous Localization and Mapping (SLAM)-Toolbox [63], this enables 2D mapping, localization, and path planning within the simulated indoor environments. Inter-process and inter-container communication is handled via FastDDS, enabling distributed communication between the detection, mapping, and exploration workspaces.

Open-vocabulary object detection is realized using YOLO-E [42], implemented via the Ultralytics framework [88]. Given a text prompt, the detector produces object hypotheses with bounding boxes and confidence scores, and provides instance-level masks that are used to localize candidate objects for subsequent verification and fusion.

Semantic similarity scoring between visual observations and user-defined text queries is performed using BLIP-2 [27], accessed through the LAVIS library [89]. This VLM provides

cosine similarity scores that are used to populate the semantic value map and guide exploration toward semantically relevant regions.

Persistent 3D semantic mapping and memory generation are handled by OpenFusion [35], which is wrapped as a dedicated ROS2 node. OpenFusion incrementally fuses RGB-D observations into a global semantic point cloud representation, serving as the multi-episode persistent memory component of the proposed system.

All experiments are conducted on the following configuration:

- **PC Workstation:**
  - **CPU:** AMD Ryzen 9 5950X (16 cores, 32 threads)
  - **Motherboard:** Gigabyte B550 Gaming X V2
  - **GPU:** NVIDIA GeForce RTX 4090 (ASUS TUF Gaming OC, 24 GB VRAM)
  - **RAM Memory:** 64 GB Corsair Vengeance LPX DDR4

4.4 Evaluation Pipeline

Due to using IsaacSim as the simulation platform instead of HabitatSim [41], a custom evaluation pipeline is implemented to compute standard object-goal navigation metrics such as SR and SPL [85]. This ensures comparability with prior work such as VLFM [9] and OneMap [16], which rely on these widely adopted metrics in embodied AI research. The pipeline operates on recorded navigation trajectories and semantic maps generated during each exploration episode and is illustrated in Figure 14.

The primary evaluation metrics considered in this work are SR and SPL [9, 16, 19]. Computing SPL requires knowledge of the geodesic shortest path from the robot's initial pose to the nearest instance of the target object. Since all experiments are conducted in a custom IsaacSim-based environment with uniformly scaled Matterport3D scenes, the built-in shortest-path planner provided by HabitatSim cannot be directly used.

Instead, the evaluation pipeline is composed of two main stages: (a) semantic environment reconstruction, and (b) geodesic shortest-path computation based on the reconstructed map.

As described in Chapter 3 Section 3.3, OpenFusion is employed to reconstruct a 3D semantic point cloud from the recorded RGB-D observations [35]. The Matterport3D category set is used as the closed set of semantic classes during evaluation [46]. For each episode, the stored semantic point cloud is filtered according to the target object category specified by the evaluation protocol. All clusters corresponding to the target class are extracted, and their 3D centroids are computed as candidate goal locations. See Figure 36 in Appendix B for example semantic pointclouds generated during evaluation.

The geodesic shortest path is then computed from the robot's initial pose to the nearest target centroid using the global planner of the ROS2 Navigation2 stack [76]. Path planning is performed on the 2D occupancy grid generated by SLAM Toolbox [63], ensuring consistency with the localization and navigation setup used during exploration. The resulting shortest path is compared to the executed trajectory to compute SR and SPL for each episode.

In contrast to the evaluation procedure of the Habitat Navigation Challenge 2023 [85], which relies on HabitatSim's internal shortest-path computation, the proposed evaluation pipeline is fully compatible with IsaacSim and the employed ROS2-based navigation stack. This enables a fair comparison against prior methods while preserving identical metric definitions and avoiding bias introduced by simulator-specific planners.

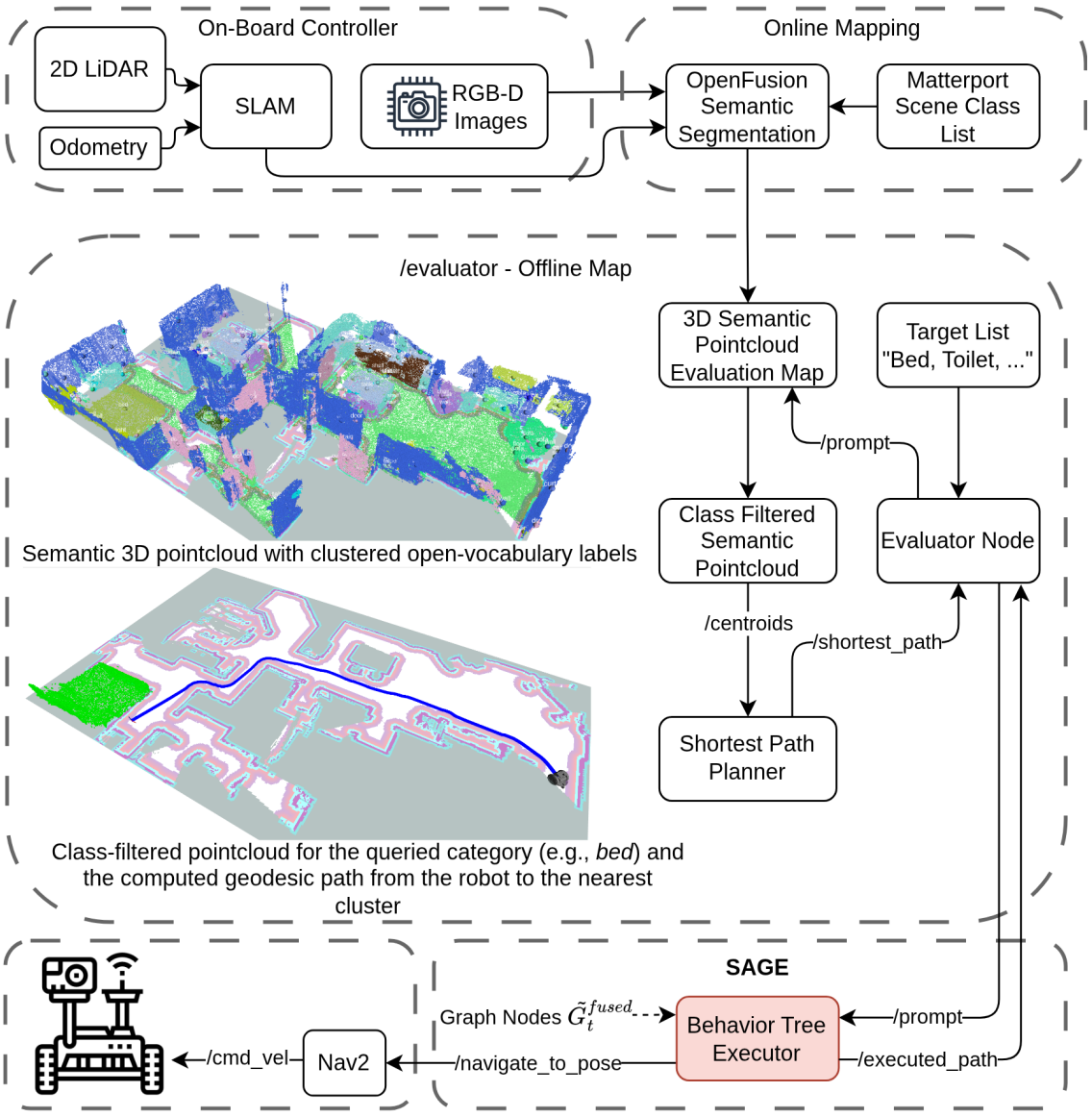


Figure 14: Evaluation pipeline used to compute SR and SPL. Recorded RGB-D observations are converted into a semantic point cloud using OpenFusion, target object clusters are extracted, and the geodesic shortest path is computed via the ROS2 Navigation2 global planner for metric evaluation [76, 85].

4.5 Evaluation Metrics

This section defines the evaluation metrics used to assess the performance of the proposed semantic exploration framework and to ensure comparability with prior work in object-goal

navigation. The selected metrics jointly evaluate the quality of semantic reasoning and the reliability of perception-driven decision making. Each metric is explicitly assigned to a corresponding experiment, such that the experimental results directly address and answer the research questions defined in Chapter 1.

Experiment 1: Overall Performance Benchmarking

The first experiment establishes a performance baseline by comparing the proposed hybrid semantic exploration framework against state-of-the-art methods on standard object-goal navigation metrics. This experiment addresses RQ1 by quantifying the overall effectiveness of the integrated exploration-memory approach in realistic indoor environments. Results are compared against representative baselines including VLFM [9], VLMaps [18], OneMap [16], and Pigeon [19]. Overall performance is evaluated using SR and SPL, which are widely adopted metrics in embodied AI research [85] and adopted by comparable prior work [9, 16].

In this experiment, an episode is defined as a single object-goal navigation task from the HM3Dv2 validation split, in which the robot is tasked with locating a specified target object within a Matterport3D-derived indoor scene [85].

The SR metric measures the proportion of successful episodes in which the robot reaches the queried target object and is defined in Equation 40,

$$\mathrm{SR} = \frac{1}{N} \sum_{i=1}^{N} S_i \tag{40}$$

where $S_i = 1$ if the target object is successfully reached in episode i, and 0 otherwise; N denotes the total number of evaluated episodes.

Navigation efficiency is quantified using SPL, which compares the geodesic shortest path to the actual path executed by the robot. The metric penalizes unnecessary detours and is defined only for successful episodes in Equation 41,

$$\mathrm{SPL} = \frac{1}{N} \sum_{i=1}^{N} S_i \cdot \frac{l_i}{\max(p_i, l_i)} \tag{41}$$

where l_i is the geodesic shortest path length to the target, p_i is the executed path length, and S_i indicates task success for episode i. Following the standard definition of SR and SPL in embodied navigation benchmarks [86], results are aggregated across the N evaluated episodes by the arithmetic mean. Using the mean is consistent with prior work that reports SR and SPL as mean performance over an evaluation split and therefore enables direct comparison to reported baseline results [9, 16, 85].

An episode is considered successful, if the target object is visible in the RGB image captured by the robot's onboard camera, as verified through post-hoc inspection of the recorded observations. The geodesic shortest path l_i is computed from the robot's initial pose to the nearest instance of the target object. In contrast to HabitatSim-based evaluations, success is not restricted to predefined viewpoint tolerances, but instead includes any robot pose from which the target object is visually observable. All reported results were manually verified to

confirm correct target visibility and episode termination. For every episode, the target object class is selected from the HM3Dv2 annotated category set [86].

Experiment 2: Exploration-Memory Fusion Weighting

This experiment is designed to address RQ2 by systematically analyzing how the balance between live semantic exploration and persistent 3D semantic memory influences navigation performance and behavioral stability. By varying the relative contribution of exploration-driven and memory-driven cues during graph node fusion, the experiment evaluates the impact of different fusion configurations on task success, navigation efficiency, and long-horizon search behavior. The primary objective is to identify a weighting configuration that provides an optimal trade-off between reactivity to newly observed information and stability derived from accumulated semantic memory.

The fusion behavior is controlled by the exploration-memory weighting parameter

$$\lambda_{\text{exploration}} \in [0, 1],$$

which determines the relative influence of frontier-based exploration and memory-driven exploitation during graph node fusion. A value of $\lambda_{\text{exploration}} = 0$ corresponds to purely memory-driven behavior, while $\lambda_{\text{exploration}} = 1$ corresponds to purely exploration-driven behavior. The parameter is varied in discrete increments of 0.2. For each weighting configuration, multiple navigation episodes are executed, and performance is evaluated using the standard object-goal navigation metrics SR and SPL (see Equation 40 and Equation 41).

For Experiments 5.2 and 5.3, metrics are aggregated in two stages. Within each multi-object episode (fixed scene, starting pose and hyperparameters), SR and SPL are first computed per target query and then averaged across the Q sequential queries of that episode (see Equation 42),

$$\overline{\mathrm{SR}}_e = \frac{1}{Q}\sum_{q=1}^{Q} \mathrm{SR}_{e,q}, \qquad \overline{\mathrm{SPL}}_e = \frac{1}{Q}\sum_{q=1}^{Q} \mathrm{SPL}_{e,q} \tag{42}$$

yielding one episode-level performance value per configuration. This within-episode mean is appropriate because all queries share the same environment state and initial conditions, and it matches the standard averaging implicit in the definitions of SR and SPL for single-object episodes, as defined in Metrics of Experiments 4.5.

Across the set of episode-level values $\{\overline{\mathrm{SR}}_e\}_{e=1}^{N}$ and $\{\overline{\mathrm{SPL}}_e\}_{e=1}^{N}$ for a given hyperparameter setting, we report the median together with the interquartile range $\mathrm{IQR} = q_{0.75} - q_{0.25}$. The median summarizes central tendency, while the Interquartile Range (IQR) quantifies dispersion over the central 50% of episode-level outcomes. This yields a compact summary of performance variation across different scenes and starting conditions.

To ensure consistency and reproducibility across configurations, Experiment 2 follows a fixed hierarchical evaluation protocol. Each episode is defined by a unique combination of scene, starting position, and exploration-memory weighting, and consists of multiple sequential object queries executed on a persistent semantic map without resetting the environment or clearing the accumulated memory.

Evaluation Dimension	Configuration
Scenes	6 Matterport3D scenes
Starting positions per scene	2
Multi-object episodes per starting position	6 (one per weighting)
Prompts per multi-object episode	5 object queries
Exploration-memory weight	Unique value per episode (0.0, 0.2, 0.4, 0.6, 0.8, 1.0)
Semantic map reset	No (persistent across prompts)

Table 9: Hierarchical evaluation structure used in Experiment 2. Each episode corresponds to a distinct exploration-memory weighting configuration and contains multiple sequential object queries executed on a persistent semantic map.

The resulting evaluation scale for this experiment is summarized in Table 10. In total, 72 multi-object episodes are executed, yielding 345 individual object-level queries across all weighting configurations.

Statistic	Value
Multi-object episodes	72
Total object queries (prompts)	345

Table 10: Evaluation scale for Experiment 2 (Exploration-Memory Fusion Weighting).

In contrast to Experiment 1, which evaluates single-object navigation episodes, Experiment 2 employs multi-object search episodes to explicitly assess the benefit of persistent semantic memory across sequential object queries. Within a single episode, the robot is required to locate multiple target objects in sequence while maintaining a shared semantic map, reflecting realistic long-horizon search scenarios in which previously acquired semantic information can be reused to inform subsequent navigation decisions.

Success for an individual target is defined identically to Experiment 1, and SPL is computed using the geodesic shortest path from the robot's pose at the start of each sub-task to the nearest instance of the corresponding target object.

For each episode, five target objects are sampled from the HM3Dv2 annotated category set [86] and queried in a fixed random order. A global random seed of 42 is used to ensure reproducibility across different exploration-memory weighting configurations. All evaluations are conducted on the HM3Dv2 validation split, using six Matterport3D scenes with two distinct starting positions per scene.

Experiment 3: Sensitivity to Semantic Map Granularity

This work employs a 3D semantic mapping approach based on OpenFusion [35], which fuses VLM embeddings into a volumetric representation. The granularity of semantic retrieval di-

rectly influences the quality of the resulting semantic map, with coarser retrievals introducing increased noise and ambiguity.

Granularity is controlled via the retrieval depth parameter top-k, which retrieves the top k most similar semantic regions from the SEEM [53] embedding dictionary within OpenFusion [35]. Lower top-k values yield sharper but potentially incomplete semantic representations, while higher top-k values produce denser semantic maps at the cost of increased noise. The top-k parameter affects the semantic retrieval stage prior to fusion and thus indirectly controls the density of the semantic map [35].

To counteract noise introduced by higher top-k values, the exploration weight $\lambda_{\text{exploration}}$ is increased across experimental configurations, shifting trust from memory-based semantic cues toward frontier-driven exploration. The top-k parameter is systematically varied from 5 to 25 in increments of 5. For each configuration, multiple navigation episodes are executed in the HM3Dv2 validation scenes, and performance is evaluated using SR and SPL.

To isolate the effect of semantic granularity, the starting pose is fixed across all runs. For each top-k value, two weighting configurations are evaluated: (a) a memory-dominant setting ($\lambda_{\text{exploration}} = 0$) and (b) an exploration-dominant configuration using the optimal weighting identified in Experiment 2 (RQ2). This design enables a direct assessment of whether increased reliance on frontier-based exploration can compensate for semantic noise introduced by higher top-k values. The resulting evaluation scale for Experiment 3 is summarized in Table 11.

Statistic	**Value**
Multi-object episodes	60
Total object queries (prompts)	324

Table 11: Evaluation scale for Experiment 3 (Sensitivity to Semantic Map Granularity).

Experiment 4: Evaluation of Multi-Source Detection Fusion

This experiment evaluates the effectiveness of the multi-source semantic fusion strategy proposed in Chapter 3 Section 3.5. Specifically, it studies how Noisy-OR fusion affects the precision-recall trade-off and the prevalence of false positives under higher detection thresholds during exploration, thereby addressing RQ4. Following prior work on open-vocabulary semantic perception, detection performance is evaluated using confusion-matrix-based metrics such as precision, recall, and F1 Score (F1)-score, which are commonly employed to assess semantic retrieval quality in open-vocabulary settings [42, 54, 72].

Each episode is manually annotated based on the executed navigation trajectory, the recorded YOLO-E detection overlay, and system logs, with a semantic outcome describing whether the robot correctly reached the target object, detected the wrong object, or observed the target but ignored it. Failure modes that are unrelated to the detection process itself, such as exploration timeouts, lack of motion, or insufficient proximity to the target, are excluded from the detection evaluation and treated as neutral cases.

Label	Condition	Interpretation
TP	Successful target reach with $S \geq \tau$	Correct detection and navigation to the target object.
FP	Object mismatch with $S \geq \tau$	Robot selects a visually or semantically similar but incorrect object.
FN	Threshold rejection	Target object is observed but not selected because all detection scores remain below τ.
TN	Target not observed (timeout / no motion)	The target object was never within the camera field of view during the episode (e.g., exploration timeout or no motion). Consequently, no positive detection decision is triggered and no false positive occurs.

Table 12: Abstracted confusion-matrix mapping for detection robustness evaluation.

Four fusion strategies are evaluated: (a) detector-only (YOLO-E), (b) detector + semantic similarity (BLIP-2), (c) detector + memory confidence, and (d) the full Noisy-OR fusion strategy combining all three sources. All fusion strategies must contain the detector source, as the other components are complementary and cannot independently trigger detections, only reinforce them. Across each fusion strategy, all experiments are executed using a fixed detection threshold of $\tau = 0.8$, which defines the robot's online decision-making behaviour and serves as the basis for manual annotation. The resulting detection confidences are then reused in a post-hoc evaluation to compute confusion matrices and classification metrics for additional thresholds $\tau = 0.5$ and 0.6.

Precision in this context measures the proportion of correct detections among all positive detections made by the robot and is defined in Equation 43:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{43}$$

where TP denotes true positives, meaning the object was correctly detected, and FP denotes false positives, meaning an incorrect object was detected. In this context, precision quantifies the system's ability to avoid incorrect detections when making a positive identification of the target object. Recall measures the proportion of actual target objects that were correctly detected and is defined in Equation 44,

$$\text{Recall} = \frac{TP}{TP + FN} \tag{44}$$

where FN denotes false negatives, meaning the target object was observed but not selected because all detection confidences remained below the decision threshold τ. In this context, recall quantifies the system's sensitivity to correctly identifying the target object when it is present in the environment. Importantly, false negatives in this experiment exclusively arise from conservative thresholding decisions and do not indicate a failure of the perception

pipeline to observe the target object. The F1-score is the harmonic mean of precision and recall, providing a balanced measure of detection performance, and is defined in Equation 45.

$$\mathrm{F1} = 2 \cdot \frac{\mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}} \tag{45}$$

A high F1-score indicates that the system effectively balances precision and recall, achieving both accurate and sensitive detection of the target object. The False Positive Rate (FPR) quantifies the proportion of incorrect detections among all negative cases and is defined in Equation 46,

$$\mathrm{FPR} = \frac{FP}{FP + TN} \tag{46}$$

where TN denotes true negatives, corresponding to episodes in which no detection decision is evaluated due to exploration-related failure modes or insufficient observability. TN corresponds to neutral episodes in which no detection decision was made and is included solely for the computation of the false-positive rate. Note that TN is defined at the episode level and corresponds to cases without target observability; thus, the reported FPR should be interpreted as the false-commit rate under non-observability rather than a frame-level false positive rate.

The confusion matrix for single-source and multi-source fusion is primarily reported for the operational threshold $\tau = 0.8$, which was empirically selected to provide a balanced trade-off between sensitivity and specificity in preliminary tests. In addition, precision–recall curves are generated by sweeping the decision threshold τ from 0 to 1 in increments of 0.01 using the same annotated episodes. This threshold sweep enables an analysis of detection robustness across different operating points without requiring additional experimental runs.

Experiment 5: Real-World System Performance

This experiment defines a deployment-oriented evaluation protocol for assessing the real-world performance of the proposed semantic exploration framework on a physical mobile robot platform. The objective is to characterize system robustness, efficiency, and deployability in realistic indoor environments, thereby addressing RQ5.

The evaluation focuses on monitoring computational resource utilization, including CPU, GPU, and RAM usage, as well as inference latency and effective FPS of the perception and mapping components during live operation. Each component's computational load is logged to enable fine-grained performance analysis.

In addition, the semantic map voxel resolution in OpenFusion is systematically reduced to analyze the trade-off between GPU memory consumption and runtime performance. This analysis is intended to assess the feasibility of deploying the system on resource-constrained robotic platforms.

5 Results and Discussion

This chapter answers the research questions defined in Chapter 1 by evaluating the proposed multi-object, open-vocabulary exploration framework in a sequence of experiments. Experiment 1 addresses (RQ1) through a benchmark-style comparison on HM3Dv2. Experiments 2–4 provide ablations that isolate key design choices of the system: (i) the exploration-memory weighting (RQ2), (ii) the semantic memory granularity via OpenFusion retrieval depth (RQ3), and (iii) multi-source semantic fusion for detection robustness (RQ4). Finally, Experiment 5 evaluates computational footprint and real-world feasibility (RQ5) under deployment conditions.

5.1 Experiment 1: Benchmarking on Matterport Scenes

This experiment evaluates the navigation success and efficiency of the proposed framework on HM3Dv2 [86] and compares it to representative state-of-the-art methods. Table 13 summarizes SR and SPL for prior work and for SAGE. Note that the experimental setup is not identical to all baselines, as the evaluation is performed in Isaac Sim with the pipeline described in Chapter 4, whereas several prior results are reported under Habitat-based evaluation protocols. Therefore, the comparison should be interpreted as an indicative reference rather than a strictly controlled re-evaluation.

Within this evaluation setting, SAGE achieves a mean SPL of 63.6% and a mean SR of 77.5%. Compared to the next best reported SPL in Table 13, this corresponds to an increase of 26.2 percentage points in SPL, while maintaining competitive success rates.

These results establish a strong reference point for the subsequent experiments, in which individual components of the framework and key hyperparameters are analyzed in isolation. All following ablation studies therefore build upon the configuration evaluated in this benchmark and aim to explain which design choices contribute most to the observed performance gains.

Figure 15 illustrates the evaluation setup for a representative episode by visualizing the executed trajectory and the corresponding geodesic shortest path used in the SPL computation. The associated detection overlay and further qualitative examples for additional prompts are provided in Appendix B.

5.2 Experiment 2: Impact of Exploration-Memory Weighting

Experiment 2 addresses RQ2 by analyzing how the weighting between live exploration signals and accumulated semantic memory affects navigation performance and failure characteristics.

Approach	TRAINING		HM3DV2	
	Locom.	Sem.	SPL ↑	SR ↑
PIRLNav [26]	✓	✓	27.1	64.1
ZSON [11]	✓	✓	12.6	25.5
ESC [10]	✗	✗	22.3	39.2
VLFM [9]	✓	✗	30.4	52.5
OneMap [16]	✗	✗	<u>37.4</u>	55.8
PIGEON-ZeroShot [19]	✓	✓	36.8	**79.2**
SAGE (this work)	✗	✗	**63.6**	<u>77.5</u>

Table 13: Quantitative comparison of navigation performance on HM3Dv2.

Let $\lambda_{\text{exploration}} \in [0, 1]$ denote the fusion weight that increases reliance on reactive exploration cues (frontier-based semantic scoring). Figures 16a and 16b report SR and SPL as a function of $\lambda_{\text{exploration}}$.

The results indicate that extreme configurations are suboptimal. For small $\lambda_{\text{exploration}}$ (memory-dominant behavior), performance becomes sensitive to noisy or stale semantic hypotheses in the map. For large $\lambda_{\text{exploration}}$ (exploration-dominant behavior), the agent is more reactive but tends to incur redundant motion and observations because previously acquired knowledge is underutilized. In the evaluated scenes, intermediate weights yield the most stable trade-off between reactivity and exploitation of accumulated information.

To contextualize these trends, Figures 17 and 18 summarize failure-mode distributions across weight configurations. Across all settings, misdetections remain the dominant failure class, primarily driven by premature commitment to visually or semantically similar distractors (`Stopped at wrong object`). Failure cases in which the target is not observed (`Did not see goal`) are attributable to incomplete coverage or occlusion, whereas `Ignored target` cases are consistent with conservative decision thresholding under ambiguity.

Figures 18a and 18b compare two representative operating points, $\lambda_{\text{exploration}} = 0.6$ and $\lambda_{\text{exploration}} = 0.4$. At $\lambda_{\text{exploration}} = 0.6$, the system relies more strongly on reactive exploration cues, which reduces over-commitment to the semantic memory but may still produce misdetections when semantically similar distractors are encountered. At $\lambda_{\text{exploration}} = 0.4$, the policy remains reactive while placing relatively more trust in memory cues. This setting benefits from accumulated evidence when frontier scores are uncertain, while still allowing new observations to correct spurious hypotheses. Empirically, this balance reduces premature commitment to incorrect targets.

At the extremes, pure memory-driven behavior ($\lambda_{\text{exploration}} = 0$) and pure exploration-driven

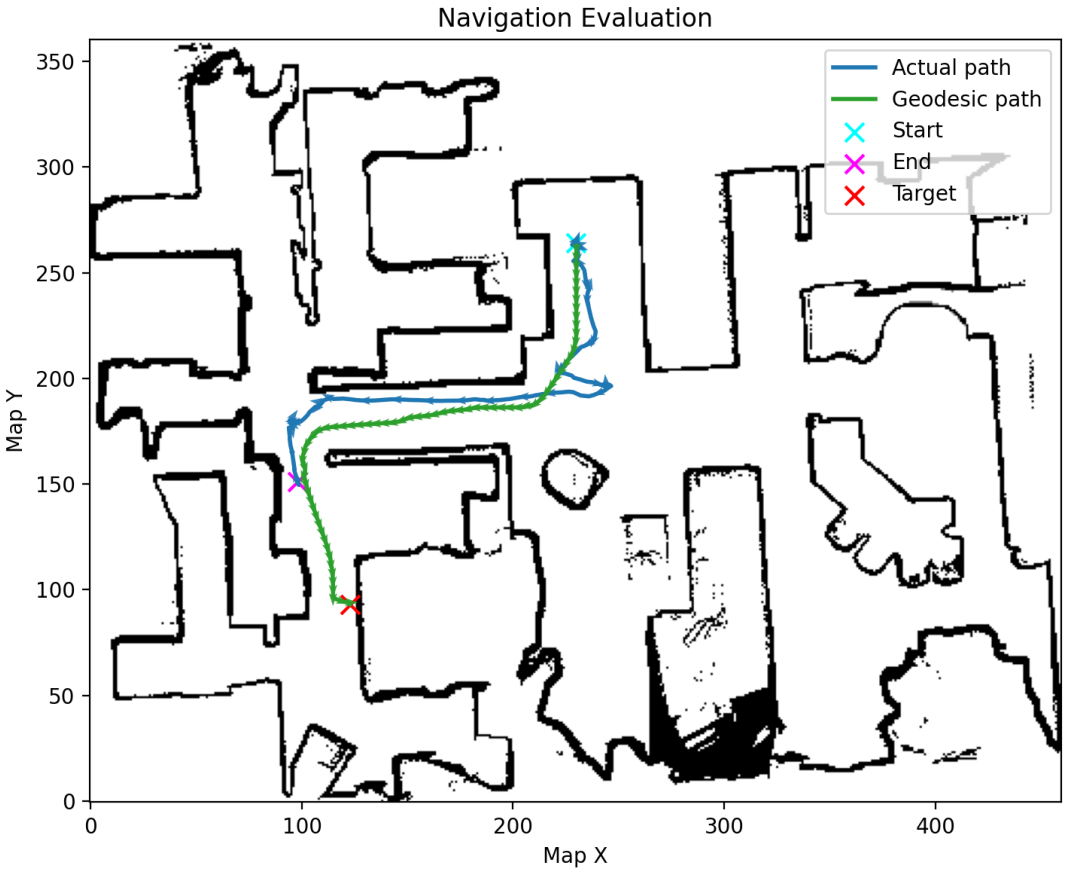
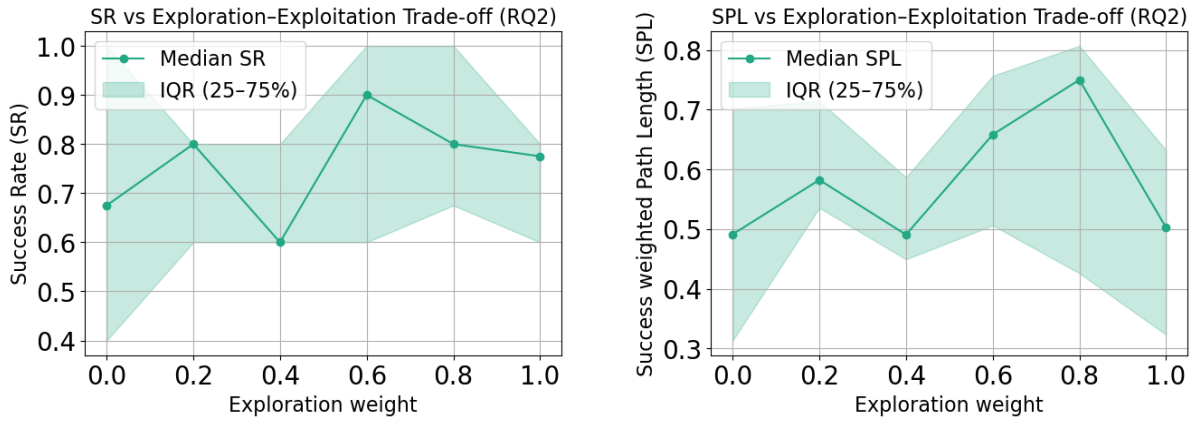


Figure 15: Example trajectory visualization for Scene `00848-ziup5kvtCCR` and target class `bed`. The plot compares the executed path with the geodesic shortest path used to compute SPL. Start, end, and target positions are indicated; the corresponding detection overlay is provided in Appendix B.



(a) SR (b) SPL

Figure 16: Navigation performance as a function of the exploration-memory weighting in Experiment 2. SR captures task success, while SPL reflects navigation efficiency conditioned on success.

behavior ($\lambda_{\text{exploration}} = 1$) exhibit similarly low SPL, indicating inefficient navigation in both cases. Without sufficient exploration, memory-dominant behavior can over-commit to false

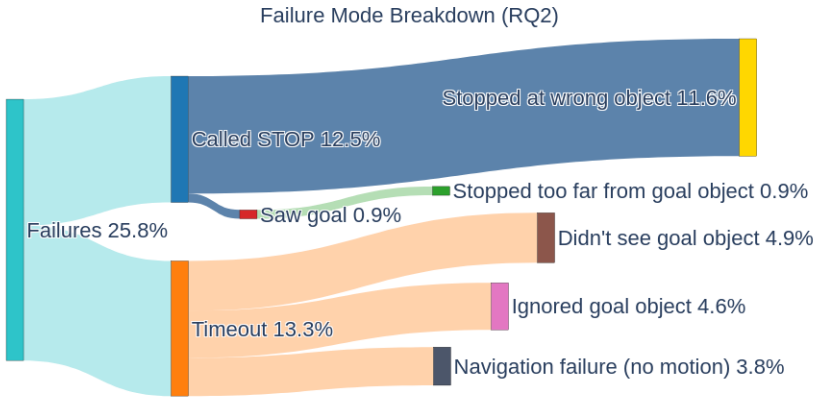
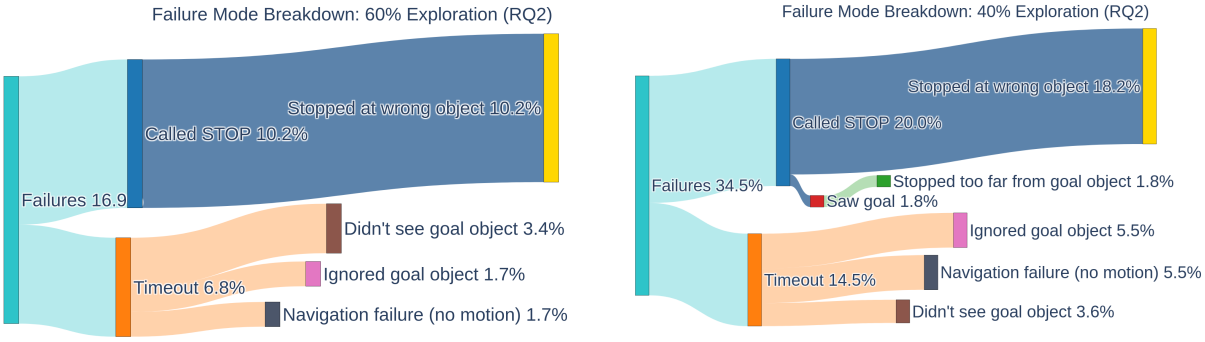


Figure 17: Aggregate failure mode distribution across all exploration-memory weight configurations for Experiment 2.



(a) Exploration weight $\lambda_{\text{exploration}} = 0.6$

(b) Exploration weight $\lambda_{\text{exploration}} = 0.4$

Figure 18: Failure mode breakdowns for two exploration-memory weighting configurations in Experiment 2. Intermediate exploration weights reduce premature commitment to noisy semantic hypotheses by combining persistent memory cues with corrective exploratory evidence.

positives stored in the map and may fail to recover when the hypothesis is wrong. Conversely, exploration-dominant behavior adapts to new evidence but does not capitalize on previously acquired information, increasing redundant search. The slightly higher SR observed for exploration-dominant settings suggests that adaptability to new observations is preferable to rigid reliance on semantic memory when semantic hypotheses are noisy.

Overall, these results indicate that the interaction between exploration and memory is nonlinear: semantic memory supports efficient long-horizon search, but excessive reliance on it increases susceptibility to uncorrected false positives and incomplete evidence. Conversely, relying exclusively on exploration leads to unnecessarily long trajectories due to limited reuse of accumulated knowledge. The most robust performance is obtained for $\lambda_{\text{exploration}} \in [0.6, 0.8]$. Consequently, $\lambda_{\text{exploration}} = 0.6$ is selected for all subsequent experiments.

5.3 Experiment 3: Sensitivity to Semantic Map Granularity

Experiment 3 addresses RQ3 by evaluating robustness to the granularity of semantic retrieval used to construct the OpenFusion-based semantic memory. The system uses OpenFusion to fuse VLM embeddings into a volumetric representation. Granularity is controlled via the retrieval depth parameter *top-k*, which retrieves the k most similar semantic entries from the SEEM embedding dictionary inside OpenFusion prior to fusion. Smaller *top-k* values yield sharper but potentially incomplete semantic hypotheses, whereas larger *top-k* values increase map density at the cost of higher noise and semantic ambiguity.

To compensate for noise under larger *top-k*, $\lambda_{\text{exploration}}$ is increased across configurations, shifting trust from memory-based cues toward frontier-driven exploration. The parameter *top-k* is varied from 5 to 25 in increments of 5, and performance is evaluated using SR and SPL with fixed starting poses to isolate semantic granularity effects.

Figure 19 illustrates the qualitative effect of different top-k values on the resulting semantic map. As expected, lower top-k values yield sparser but cleaner semantic representations, while higher top-k values produce denser maps that contain more semantic cues but also increased noise. In particular, higher top-k retrievals introduce additional false positives, as semantically similar but incorrect objects may be assigned non-negligible relevance scores. This effect is exemplified by the prompt `"couch"`, for which unrelated objects such as a stove or table are incorrectly highlighted at higher granularity levels.

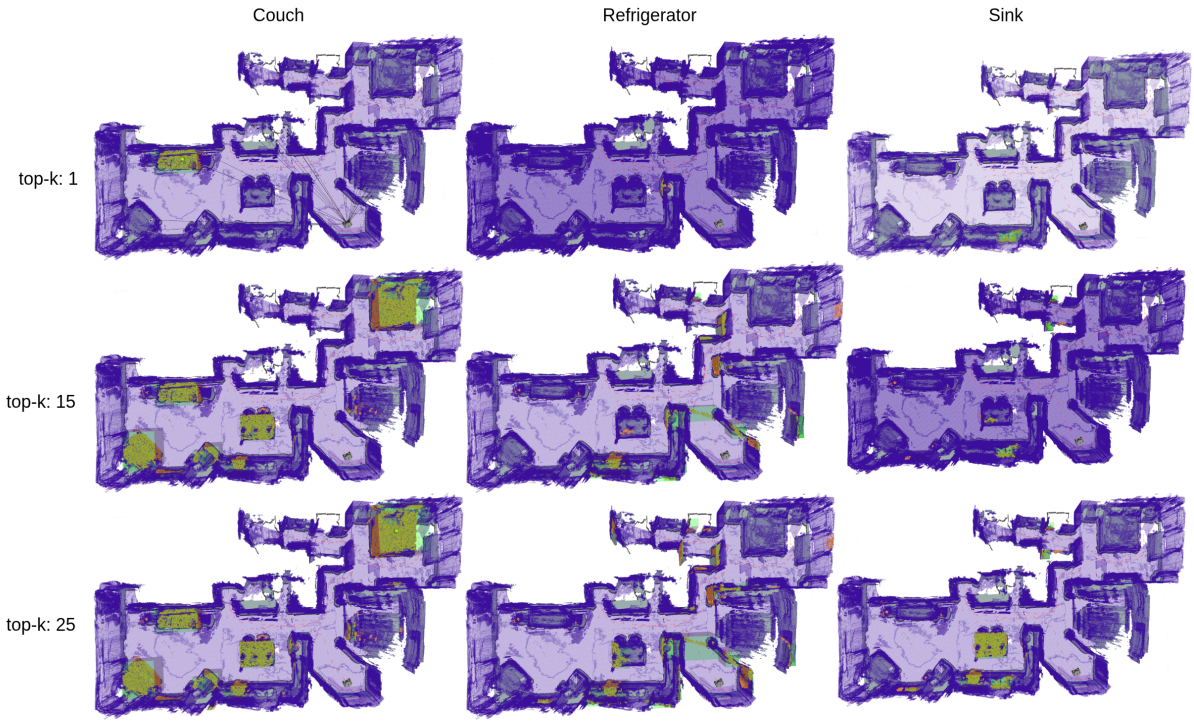


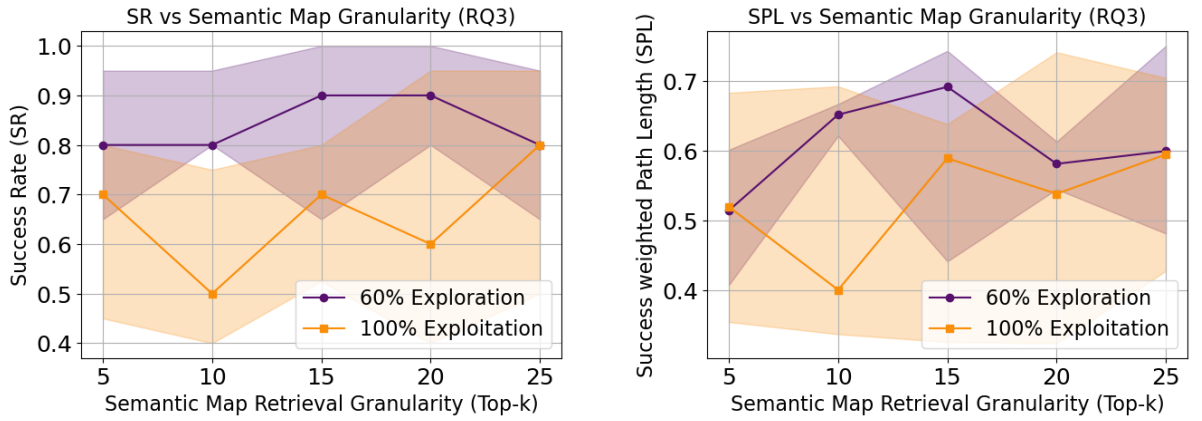
Figure 19: Qualitative effect of semantic map granularity on the OpenFusion memory. Increasing the top-k retrieval depth densifies the semantic map but also introduces additional noise and false positive associations.

Across all 60 multi-object episodes, comprising a total of 324 object-level queries, navigation

performance remains relatively stable under substantial variations in semantic map granularity. Figures 20a and 20b show the SR and SPL as a function of top-k for two exploration-memory configurations: a balanced strategy with 60% exploration and a memory-dominant strategy with 100% exploitation.

Consistent with the findings of Experiment 2, the balanced configuration (60% exploration) consistently outperforms pure exploitation across all top-k values for both SR and SPL. While the absolute best performance is achieved at a top-k value of 15, the overall performance differences across granularity levels remain modest, indicating that the proposed framework is robust to variations in semantic map density and noise.

Notably, the IQR for pure exploitation is substantially larger than for the balanced configuration. This increased variance suggests that exclusive reliance on semantic memory amplifies sensitivity to noisy semantic cues introduced at higher top-k values, whereas incorporating exploratory evidence stabilizes performance.



(a) SR as a function of semantic map granularity

(b) SPL as a function of semantic map granularity

Figure 20: Impact of semantic map retrieval granularity (top-k) on navigation performance in Experiment 3. Results are shown for a balanced exploration-memory configuration (60% exploration) and a memory-dominant configuration (100% exploitation).

To further analyze the effect of semantic map granularity on system robustness, Figure 21 compares the failure mode distributions for the balanced and pure exploitation configurations. Overall failure rates are substantially lower for the balanced configuration (18.0%) than for pure exploitation (34.2%).

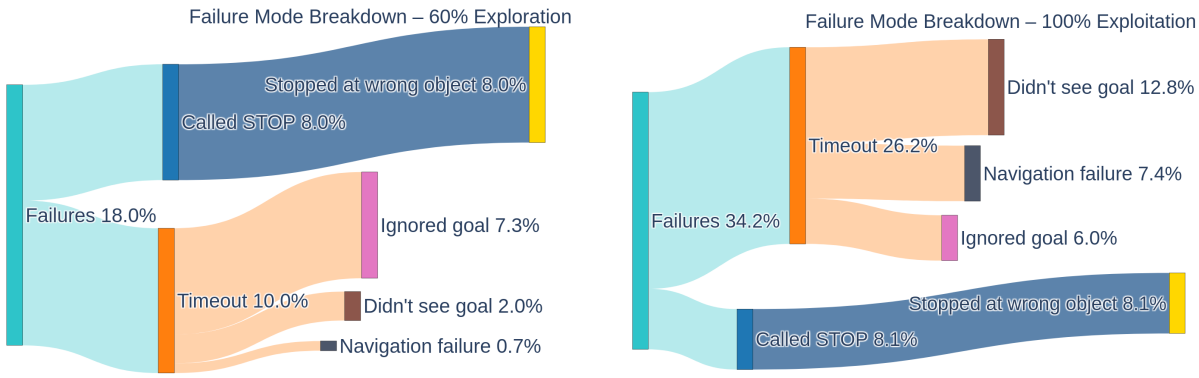
Interestingly, the dominant failure mode for both configurations is `Stopped at wrong object`, with comparable rates of 8.0% and 8.1% for balanced and pure exploitation, respectively. This consistency indicates that the detection pipeline itself behaves similarly across configurations and that misdetections are not directly caused by the exploration-memory weighting.

In contrast, pure exploitation exhibits markedly higher rates of `Did not see goal` (12.8%) and `Ignored target` (6.0%) failures compared to the balanced configuration (2.0% and 7.3%, respectively). These failures are primarily associated with insufficient environment coverage and conservative target selection in the absence of corrective exploratory behavior.

Furthermore, navigation failures occur substantially more often under pure exploitation (7.4%) than under the balanced configuration (0.7%). Navigation failures are defined as situa-

tions in which the robot becomes trapped between obstacles, no valid navigation plan can be generated, or the selected plan leads into a dead-end that moves the robot farther away from the target.

The pronounced reduction in navigation failures under the balanced configuration is attributed to the exploration component of the proposed framework, which incorporates costmap-based information to penalize frontiers located near obstacles and to bias navigation toward safer, more traversable regions. In contrast to purely similarity-driven approaches such as VLFM [9], the proposed method explicitly integrates geometric and navigational constraints into frontier scoring, thereby improving path feasibility and overall navigation robustness.



(a) Balanced configuration (60% exploration)

(b) Pure exploitation (100% exploitation)

Figure 21: Failure mode breakdowns for Experiment 3 under different exploration-memory weighting strategies. Increased reliance on semantic memory amplifies observation- and navigation-related failures, whereas balanced exploration mitigates premature commitment to noisy semantic hypotheses.

The best-performing configurations in Experiment 3 are consistent with the results of Experiment 2. In both experiments, the highest SR and SPL are achieved for top-k values in the range of 10 to 15. This alignment is expected, as Experiment 2 was conducted using a top-k value of 10, which was selected empirically during preliminary testing and used as a reasonable initial configuration for subsequent experiments.

Overall, the results show that coarse semantic retrieval can degrade performance when the policy over-trusts the memory, but that balanced exploration can mitigate these effects by re-acquiring evidence through additional viewpoints and observations.

5.4 Experiment 4: Effect of Multi-Source Semantic Fusion

A key component of the proposed framework is the multi-source fusion strategy introduced in Chapter 3 Section 3.5. Detecting target objects reliably in cluttered, real-world environments is inherently challenging due to occlusions, varying lighting conditions, and visual ambiguities. Relying on a single detection source increases susceptibility to false positives and negatives, which can mislead navigation decisions. To mitigate this, the proposed method fuses detection cues from multiple sources: (a) live detections from an open-vocabulary detector (YOLO-E [42]), (b) semantic similarity score from the VLM [27], and (c) accumulated temporal evidence

from the semantic memory (OpenFusion [35]). This experiment evaluates the effectiveness of this multi-source fusion strategy compared to single-source detection and pairwise combinations, with a Noisy-Or fusion approach as described in Chapter 3 Section 3.5. Experiment 5.4 isolates the perception and decision layer by evaluating how multi-source semantic fusion affects detection robustness, false positives, and threshold sensitivity, independent of exploration strategy.

Table 14 reports detection performance across different fusion strategies and decision thresholds. Results are shown for three operating points corresponding to increasing levels of conservativeness: a low threshold ($\tau = 0.5$), a medium threshold ($\tau = 0.6$), and a high threshold ($\tau = 0.8$). For each threshold, four fusion variants are compared: single-source detection using YOLO-E, detection combined with VLM-based semantic similarity, detection combined with semantic memory evidence, and the proposed multi-source fusion integrating all three cues via a Noisy-Or formulation.

Fusion Variant	τ	$P \uparrow$	$R \uparrow$	$F1 \uparrow$	$FPR \downarrow$	TP	FP	FN	TN
Single source detection	0.5	**0.910**	0.909	0.910	**0.364**	528	52	53	91
Detection + VLM score	0.5	0.903	**0.928**	**0.915**	0.406	539	58	42	85
Detection + Memory	0.5	<u>0.909</u>	0.914	<u>0.912</u>	<u>0.371</u>	531	53	50	90
Multi-source Fusion	0.5	0.906	<u>0.917</u>	<u>0.912</u>	0.385	533	55	48	88
Single source detection	0.6	**0.914**	0.892	0.902	**0.343**	518	49	63	94
Detection + VLM score	0.6	<u>0.910</u>	**0.910**	**0.910**	<u>0.364</u>	529	52	52	91
Detection + Memory	0.6	<u>0.910</u>	<u>0.905</u>	<u>0.908</u>	<u>0.364</u>	526	52	55	91
Multi-source Fusion	0.6	<u>0.910</u>	**0.910**	**0.910**	<u>0.364</u>	529	52	52	91
Single source detection	0.8	**0.942**	0.636	0.759	**0.161**	375	23	215	120
Detection + VLM score	0.8	0.912	<u>0.876</u>	<u>0.894</u>	0.343	509	49	72	94
Detection + Memory	0.8	<u>0.913</u>	0.799	0.852	<u>0.308</u>	464	44	117	99
Multi-source Fusion	0.8	<u>0.915</u>	**0.890**	**0.902**	0.343	525	49	65	94

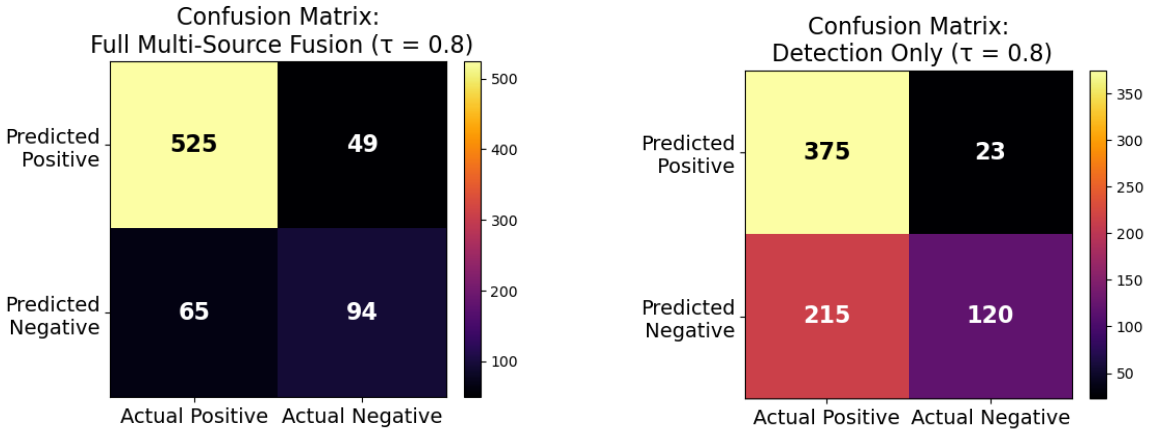
Table 14: Detection performance metrics across fusion strategies and thresholds. Arrows indicate whether higher (↑) or lower (↓) values are better. For each threshold τ, the best-performing value per metric is highlighted in bold and the second-best value is underlined.

At the low threshold ($\tau = 0.5$), all variants achieve high recall (0.909–0.928), indicating that most true target observations are accepted. Differences between fusion strategies are relatively small in this regime. Single-source detection achieves the highest precision and the lowest false-positive rate, while incorporating semantic similarity improves recall at the cost of increased false positives. The multi-source fusion yields a balanced trade-off, with slightly reduced precision compared to detection-only but improved recall relative to the single-source baseline.

At the medium threshold ($\tau = 0.6$), performance across all fusion variants becomes more balanced. Precision values cluster around 0.91, and recall differences are reduced. In this regime, both the detection+VLM variant and the full multi-source fusion achieve the highest F1-scores, indicating an effective balance between sensitivity and specificity. The corresponding confusion-matrix counts show that fusion-based approaches reduce false negatives compared to detection-only, while maintaining comparable false-positive rates.

At the high threshold ($\tau = 0.8$), clear differences between fusion strategies emerge. Single-source detection attains the highest precision (0.942) and the lowest false-positive rate, but suffers from a substantial drop in recall (0.636), resulting in a large number of false negatives. In contrast, all fusion-based variants significantly improve recall under this conservative operating point. The proposed multi-source fusion achieves the highest recall (0.890) and F1-score (0.902), while maintaining precision above 0.915. This improvement corresponds to a pronounced reduction in false negatives compared to detection-only, with only a moderate increase in false positives.

Figure 22 provides a qualitative comparison of the detection behavior at a conservative operating point ($\tau = 0.8$) for the proposed multi-source fusion strategy and the detection-only baseline. The detection-only configuration exhibits a pronounced imbalance between precision and recall: while most predicted positives are correct, a large number of true target instances are rejected, resulting in a high false-negative count. This behavior reflects the brittleness of relying on a single detection signal under strict thresholding, where ambiguous or partially occluded targets are frequently missed.



(a) Multi-source semantic fusion ($\tau = 0.8$)

(b) Detection-only baseline ($\tau = 0.8$)

Figure 22: Confusion matrices comparing the proposed multi-source semantic fusion strategy with a detection-only baseline at a fixed decision threshold $\tau = 0.8$. While detection-only achieves high precision, it suffers from a large number of false negatives. In contrast, multi-source fusion substantially improves recall by reducing missed detections, while maintaining a comparable false-positive rate.

In contrast, the multi-source fusion strategy substantially reduces the number of false negatives by integrating complementary semantic cues from the VLM similarity score and accumulated memory evidence. The additional contextual information allows the system to recover

target instances that would otherwise fall below the detection confidence threshold. Importantly, this gain in recall is achieved without a disproportionate increase in false positives, indicating that the fusion mechanism does not merely relax the decision criterion but selectively reinforces consistent evidence across sources.

Figure 23 illustrates the precision-recall characteristics of detection-only and multi-source semantic fusion over a full sweep of the decision threshold τ. While both approaches achieve high precision at low recall levels, the detection-only baseline exhibits a steeper precision drop as recall increases, indicating reduced robustness when operating beyond conservative thresholds. In contrast, the proposed multi-source fusion maintains consistently higher precision over a broader recall range (0.6–0.9 Recall), demonstrating improved stability under varying confidence requirements.

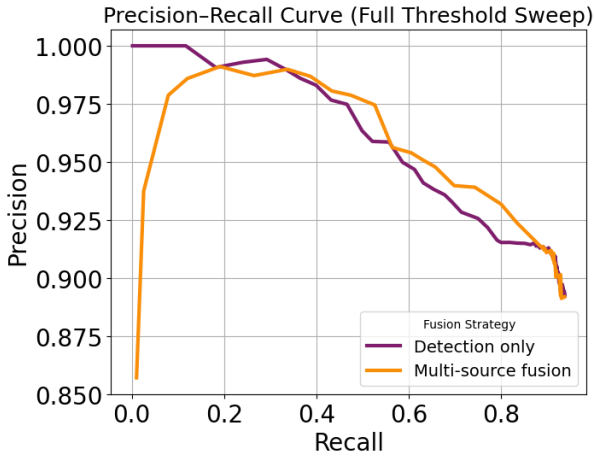


Figure 23: Precision-recall curves for detection-only and multi-source semantic fusion obtained by sweeping the decision threshold τ in Experiment 4. The multi-source fusion maintains higher precision over a wider recall range, indicating improved robustness compared to single-source detection.

This behavior reflects the complementary nature of the fused cues. When detection confidence alone is insufficient, semantic similarity and accumulated memory evidence reinforce plausible target hypotheses, allowing true positives to be recovered without a disproportionate increase in false positives. As a result, the multi-source fusion achieves a more favorable precision-recall trade-off, particularly in the mid- to high-recall regime that is critical for reliable downstream decision-making.

In the deployed behavior tree, this trade-off is explicitly exploited through a two-stage verification strategy. Candidate objects are initially detected using a permissive threshold ($\tau = 0.6$) to trigger alignment and viewpoint refinement toward the object. Final confirmation is then performed at a stricter threshold ($\tau = 0.8$), ensuring high-confidence approval before task completion. The precision-recall characteristics observed in Figure 23 support this design choice, as multi-source fusion preserves recall at lower thresholds while retaining high precision at conservative operating points.

Overall, the results demonstrate that while single-source detection performs well at permissive thresholds, it becomes increasingly brittle as the decision threshold is raised. Incorporat-

ing semantic similarity and memory evidence stabilizes detection performance across thresholds, and the proposed multi-source fusion provides the most robust trade-off between precision and recall, particularly in conservative regimes that are critical for reliable downstream navigation decisions.

5.5 Experiment 5: System Efficiency and Real-World Validation

This experiment addresses RQ5 by evaluating the computational efficiency and real-time feasibility of the proposed system in a real-world deployment setting. Figures 24 and **??** (see Appendix **??**) summarize the GPU memory consumption of the overall system and an example of a semantic Map for the given total voxel allocation, respectively.

The **SAGE** framework consists of three primary GPU-intensive components: (a) the open-vocabulary detector YOLO-E [42], (b) the VLM BLIP-2 [27] for semantic similarity scoring, and (c) the OpenFusion [35] semantic memory module for persistent 3D fusion. YOLO-E and BLIP-2 exhibit fixed GPU memory footprints of approximately 1200 MB and 2280 MB of VRAM, respectively, and do not vary during runtime. In contrast to VLFM [9], which utilizes four different VLM and detectors, which cumulatively require over 16 GB of GPU memory, **SAGE** only uses 3480 MB for the same functionality, making it more suitable for consumer-grade hardware.

In contrast, the GPU memory consumption of OpenFusion depends on the number of allocated voxels in the volumetric map. In the evaluated configuration, a total of 191,941 voxels are allocated, resulting in a GPU memory usage of approximately 6122 MB for the semantic memory. This configuration is sufficient to represent an apartment-scale environment, as illustrated in Figure **??**. Overall, the complete **SAGE** system requires approximately 9.6 GB of GPU VRAM, making it compatible with contemporary consumer-grade GPUs.

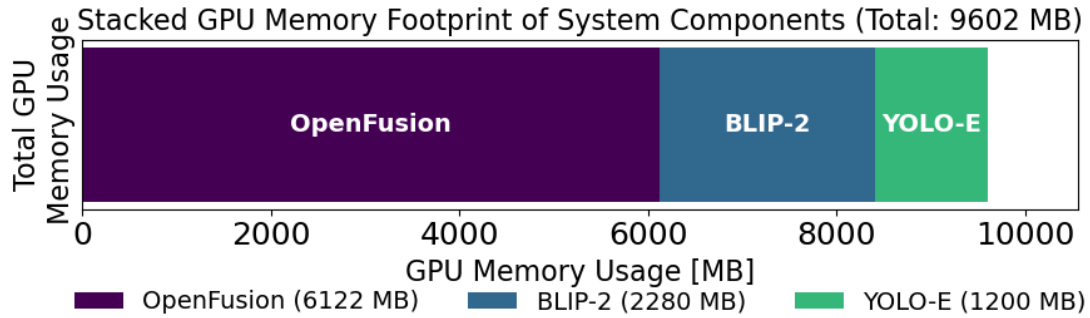


Figure 24: GPU memory consumption of the **SAGE** system during runtime, decomposed by major components.

Real-time performance is evaluated by measuring the effective processing rates of the system during deployment. Figure 25 reports the observed execution frequencies of the major processing loops. The architecture follows a deliberate multi-rate design comprising three

distinct loops: (a) a high-rate reactive exploration loop driven by the BLIP-2 ValueMap, (b) a perception loop responsible for YOLO-E detections, and (c) a low-rate persistent semantic mapping loop for OpenFusion-based fusion.

The reactive exploration loop operates at approximately 10 Hz, enabling timely responses to newly observed semantic cues. The perception loop runs at approximately 3.5 Hz, which is sufficient for stable object detection in indoor navigation scenarios. The persistent semantic mapping loop operates at a significantly lower frequency of approximately 0.36 Hz, reflecting the high computational cost of volumetric fusion. This design choice ensures that computationally expensive operations do not impede system responsiveness.

Due to the comparatively high latency of OpenFusion, the semantic memory is queried only at the beginning of a new exploration step and during candidate object confirmation. Downstream processing stages, including clustering of semantic point clouds into graph nodes, are executed at a fixed rate of 10 Hz, ensuring that higher-level decision-making components always have access to the most recent fused semantic information. The graph node fusion stage integrates signals from detection, semantic similarity for the value map, and memory at approximately 1.6 Hz during exploration and 1.7 Hz during detection, which was found to be sufficient for reliable navigation without excessive computational overhead.

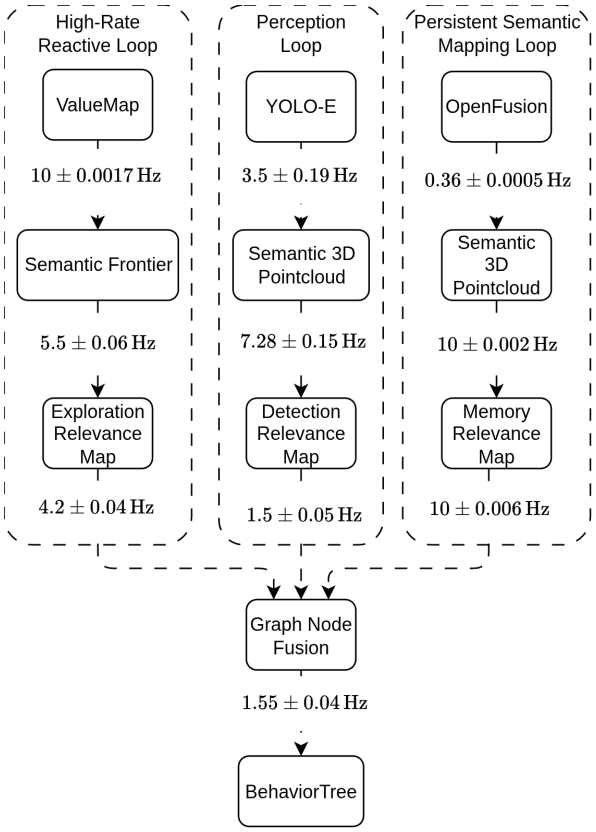


Figure 25: Observed execution frequencies of the main processing loops during real-world deployment.

6 Summary and Outlook

Open-vocabulary semantic exploration in unstructured environments represents a fundamental challenge at the intersection of robotics, computer vision, and natural language understanding. Autonomous agents must not only navigate efficiently through unknown spaces, but also reason about semantically meaningful targets specified at runtime, under conditions of partial observability, perceptual ambiguity, and limited computational resources.

Prior work in semantic exploration has primarily followed two directions. Reinforcement learning-based approaches achieve strong performance through extensive training in simulation [11, 25, 26], but often suffer from limited generalization and high training costs. In contrast, recent zero-shot methods leverage VLMs to guide exploration toward semantically relevant regions without task-specific training [9, 14]. To further improve efficiency, several frameworks incorporate persistent semantic maps to accumulate knowledge over time and reduce redundant observations [16, 18]. However, existing methods either rely exclusively on semantic maps for decision-making or omit persistent mapping entirely, which can lead to brittle behavior, inefficient exploration, or limited robustness to perceptual noise. Moreover, many approaches do not explicitly fuse multi-source semantic evidence, leaving object detection performance vulnerable to false negatives and inconsistent observations.

This thesis introduced **SAGE**, a hybrid open-vocabulary semantic exploration framework that combines frontier-based exploration with persistent semantic memory and vision-language reasoning. By integrating reactive semantic frontier scoring with accumulated semantic maps, **SAGE** balances exploration of unseen space and exploitation of previously acquired knowledge. A multi-source fusion strategy further enhances robustness by combining detection confidence, vision-language similarity, and temporal memory evidence, allowing the system to reinforce consistent semantic hypotheses across viewpoints and time.

Experiments were conducted in IsaacSim on HM3Dv2 scenes [86] using an evaluation pipeline reporting common object-goal navigation metrics (e.g., SR and SPL) while operating outside the HabitatSim stack. **SAGE** achieves strong performance across scenes and start states, and the ablation studies: (i) balancing frontier-driven exploration and memory exploitation improves stability in the presence of noisy semantic cues, (ii) costmap-informed frontier scoring reduces navigation failures in cluttered regions (see Chapter 5 Section 5.3), and (iii) multi-source semantic fusion substantially reduces false negatives in object confirmation, improving downstream decision-making despite a moderate increase in false positives.

Beyond performance, this work emphasizes system efficiency and practical deployability. Compared to vision-language frontier pipelines that instantiate multiple large perception models (e.g., VLFM [9]), **SAGE** employs a streamlined set of GPU-intensive components. The exploration-related perception modules require approximately 3.5,GB of GPU memory in the presented configuration, and with a medium-sized semantic map of roughly 200,000 voxels the

total footprint remains below 10,GB. This resource profile supports deployment on consumer-grade GPUs and enables distributed configurations in which the semantic memory can be offloaded to a separate compute unit.

Despite these results, several avenues for future work remain. First, the semantic mapping backbone could be replaced or augmented with more advanced open-vocabulary mapping frameworks that provide improved fusion quality and uncertainty handling, such as OTAS [54] or DualMap [47]. Second, the exploration-exploitation balance in **SAGE** is currently governed by a fixed hyperparameter. Adaptive strategies that adjust this balance online based on environmental complexity, task progress, or perceptual confidence could further improve robustness and efficiency. Finally, higher-level task reasoning could be incorporated by leveraging LLMs through existing ROS 2 action interfaces. Such models could generate sub-goals, adjust decision thresholds, or modulate exploration behavior dynamically, enabling more complex and long-horizon tasks to be executed autonomously.

SAGE combines frontier-based semantic exploration with persistent 3D semantic memory and multi-source confidence fusion. In the presented IsaacSim [80] evaluation on HM3Dv2 scenes [86], performance is reported using the standard object-goal navigation metrics SR and SPL [85]. The ablations indicate that (a) intermediate exploration-memory weighting reduces sensitivity to semantic noise in the semantic map, (b) costmap-informed frontier scoring reduces navigation-related failures in cluttered regions, and (c) multi-source fusion improves the precision-recall trade-off of object confirmation under conservative thresholds. In addition, GPU memory usage and measured loop frequencies are reported to characterize the computational footprint.

Bibliography

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[2] A. Topiwala, P. Inani, and A. Kathpal, Frontier based exploration for autonomous robot, 2018. DOI: 10.48550/arXiv.1806.03581. arXiv: 1806.03581 [cs.RO]. [Online]. Available: https://arxiv.org/abs/1806.03581.

[3] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in Advances in Neural Information Processing Systems, vol. 33, Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[4] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in Proc. Int. Conf. on Machine Learning (ICML), ser. Proceedings of Machine Learning Research, vol. 139, PMLR, 2021, pp. 8748–8763. [Online]. Available: https://proceedings.mlr.press/v139/radford21a.html.

[5] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, Q. Vuong, V. Vanhoucke, H. Tran, R. Soricut, A. Singh, J. Singh, P. Sermanet, P. R. Sanketi, G. Salazar, M. S. Ryoo, K. Reymann, K. Rao, K. Pertsch, I. Mordatch, H. Michalewski, Y. Lu, S. Levine, L. Lee, T.-W. E. Lee, I. Leal, Y. Kuang, D. Kalashnikov, R. Julian, N. J. Joshi, A. Irpan, B. Ichter, J. Hsu, A. Herzog, K. Hausman, K. Gopalakrishnan, C. Fu, P. Florence, C. Finn, K. A. Dubey, D. Driess, T. Ding, K. M. Choromanski, X. Chen, Y. Chebotar, J. Carbajal, N. Brown, A. Brohan, M. G. Arenas, and K. Han, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," in Proceedings of the Conference on Robot Learning, ser. Proceedings of Machine Learning Research, vol. 229, PMLR, 2023, pp. 2165–2183. [Online]. Available: https://proceedings.mlr.press/v229/zitkovich23a.html.

[6] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in Computer Vision – ECCV 2014, Cham: Springer International Publishing, 2014, pp. 740–755. DOI: 10.1007/978-3-319-10602-1_48.

[7] S. Schwaiger, L. Muster, G. Novotny, M. Schebek, W. Wöber, S. Thalhammer, and C. Böhm. "UGV-CBRN: An unmanned ground vehicle for chemical, biological, radiological, and nuclear disaster response." arXiv: 2406.14385 [cs.RO]. (2024), [Online]. Available: https://arxiv.org/abs/2406.14385 (visited on 01/02/2026).

[8] J. Chen, G. Li, S. Kumar, B. Ghanem, and F. Yu, "How to not train your dragon: Training-free embodied object goal navigation with semantic frontiers," Jul. 2023. DOI: 10.15607/RSS.2023.XIX.075.

[9] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher, "VLFM: Vision-Language Frontier Maps for Zero-Shot Semantic Navigation," in 2024 IEEE International Conference on Robotics and Automation (ICRA), 2024, pp. 42–48. DOI: 10.1109/ICRA57147.2024.10610712.

[10] K. Zhou, K. Zheng, C. Pryor, Y. Shen, H. Jin, L. Getoor, and X. E. Wang, ESC: Exploration with Soft Commonsense Constraints for Zero-shot Object Navigation, 2023. DOI: 10.48550/arXiv.2301.13166. arXiv: 2301.13166 [cs.AI]. [Online]. Available: https://arxiv.org/abs/2301.13166.

[11] A. Majumdar, G. Aggarwal, B. Devnani, J. Hoffman, and D. Batra, "ZSON: Zero-shot object-goal navigation using multimodal goal embeddings," in Advances in Neural Information Processing Systems, vol. 35, Curran Associates, Inc., 2022, pp. 32 340–32 352. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/d0b8f0c8f79d3a621af945cafb669f4b-Paper-Conference.pdf.

[12] I. Lluvia, E. Lazkano, and A. Ansuategi, "Active mapping and robot exploration: A survey," Sensors, vol. 21, no. 7, p. 2445, 2021, ISSN: 1424-8220. DOI: 10.3390/s21072445. [Online]. Available: https://www.mdpi.com/1424-8220/21/7/2445.

[13] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2002, pp. 540–545. DOI: 10.1109/IRDS.2002.1041446.

[14] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt, and S. Song, "Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 23 171–23 181.

[15] O. Alama, A. Bhattacharya, H. He, S. Kim, Y. Qiu, W. Wang, C. Ho, N. Keetha, and S. Scherer, "Rayfronts: Open-set semantic ray frontiers for online scene understanding and exploration,"
in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2025, pp. 5930–5937. DOI: 10.1109/IROS60139.2025.11245813.

[16] F. L. Busch, T. Homberger, J. Ortega-Peimbert, Q. Yang, and O. Andersson, “One map to find them all: Real-time open-vocabulary mapping for zero-shot multi-object navigation,” in Proc. IEEE Int. Conf. on Robotics and Automation (ICRA), 2025, pp. 14 835–14 842. DOI: 10.1109/ICRA55743.2025.11128393.

[17] Q. Gu, A. Kuwajerwala, S. Morin, K. M. Jatavallabhula, B. Sen, A. Agarwal, C. Rivera, W. Paul, K. Ellis, R. Chellappa, C. Gan, C. M. de Melo, J. B. Tenenbaum, A. Torralba, F. Shkurti, and L. Paull, “Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning,” in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2024, pp. 5021–5028. DOI: 10.1109/ICRA57147.2024.10610243.

[18] C. Huang, O. Mees, A. Zeng, and W. Burgard, “Visual language maps for robot navigation,” in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2023, pp. 10 608–10 615. DOI: 10.1109/ICRA48891.2023.10160969.

[19] C. Peng, Z. Zhang, C. Chi, X. Wei, Y. Zhang, H. Wang, P. Wang, Z. Wang, J. Liu, and S. Zhang, PIGEON: VLM-driven object navigation via points of interest selection, 2025. DOI: 10.48550/arXiv.2511.13207. arXiv: 2511.13207 [cs.RO]. [Online]. Available: https://arxiv.org/abs/2511.13207.

[20] P. Quin, D. Nguyen, T. Vu, A. Alempijevic, and G. Paul, “Approaches for efficiently detecting frontier cells in robotics exploration,” Frontiers in Robotics and AI, vol. 8, p. 616 470, 2021. DOI: 10.3389/frobt.2021.616470.

[21] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics. Cambridge, MA, USA: MIT Press, 2006, ISBN: 978-0-262-20162-9.

[22] S. Liu, M. Zhang, P. Kadam, and C.-C. J. Kuo, 3D Point Cloud Analysis, Deep Learning, and Explainable Machine Learning Methods. Cham: Springer, 2021, ISBN: 978-3-030-89179-4. DOI: 10.1007/978-3-030-89180-0.

[23] M. Tellaroli, M. Luperto, M. Antonazzi, and N. Basilico, “Frontier-based exploration for multi-robot rendezvous in communication-restricted unknown environments,” in 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2024, pp. 5807–5812. DOI: 10.1109/IROS58592.2024.10801321.

[24] V. S. Dorbala, J. F. Mullen, and D. Manocha, “Can an embodied agent find your “cat-shaped mug”? LLM-based zero-shot object navigation,” IEEE Robotics and Automation Letters, vol. 9, no. 5, pp. 4083–4090, 2024. DOI: 10.1109/LRA.2023.3346800.

[25] S. K. Ramakrishnan, D. S. Chaplot, Z. Al-Halah, J. Malik, and K. Grauman, “Poni: Potential functions for objectgoal navigation with interaction-free learning,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 18 890–18 900.

[26] R. Ramrakhya, D. Batra, E. Wijmans, and A. Das, "Pirlnav: Pretraining with imitation and RL finetuning for objectnav," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 17 896–17 906.

[27] J. Li, D. Li, S. Savarese, and S. Hoi, "BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," in Proceedings of the 40th International Conference on Machine Learning, ser. Proceedings of Machine Learning Research, vol. 202, PMLR, 2023, pp. 19 730–19 742. [Online]. Available: https://proceedings.mlr.press/v202/li23q.html.

[28] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, J. Zhu, and L. Zhang, Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection, 2024. DOI: 10.48550/arXiv.2303.05499. arXiv: 2303.05499 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2303.05499.

[29] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," in Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Oct. 2023, pp. 4015–4026.

[30] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017.

[31] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," CoRR, vol. abs/1810.04805, 2018. arXiv: 1810.04805. [Online]. Available: http://arxiv.org/abs/1810.04805.

[32] L. H. Li, P. Zhang, H. Zhang, J. Yang, C. Li, Y. Zhong, L. Wang, L. Yuan, L. Zhang, J.-N. Hwang, K.-W. Chang, and J. Gao, "Grounded language-image pre-training," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 10 965–10 975.

[33] D. Jiang, Y. Liu, S. Liu, J. Zhao, H. Zhang, Z. Gao, X. Zhang, J. Li, and H. Xiong, From CLIP to DINO: Visual encoders shout in multi-modal large language models, arXiv preprint, 2024. DOI: 10.48550/arXiv.2310.08825. arXiv: 2310.08825 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2310.08825.

[34] D. S. Chaplot, D. Gandhi, A. Gupta, and R. Salakhutdinov, "Object goal navigation using goal-oriented semantic exploration," in Advances in Neural Information Processing Systems, vol. 33, Curran Associates, Inc., 2020, pp. 4247–4258. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/2c75cf2681788adaca63aa95ae028b22-Paper.pdf.

[35] K. Yamazaki, T. Hanyu, K. Vo, T. Pham, M. Tran, G. Doretto, A. Nguyen, and N. Le, "Open-Fusion: Real-time Open-Vocabulary 3D Mapping and Queryable Scene Representation," in 2024 IEEE International Conference on Robotics and Automation (ICRA), 2024, pp. 9411–9417. DOI: 10.1109/ICRA57147.2024.10610193.

[36] K. M. Jatavallabhula, A. Kuwajerwala, Q. Gu, M. Omama, T. Chen, A. Maalouf, S. Li, G. Iyer, S. Saryazdi, N. Keetha, A. Tewari, J. B. Tenenbaum, C. M. de Melo, M. Krishna, L. Paull, F. Shkurti, and A. Torralba, "Conceptfusion: Open-set multimodal 3d mapping," in Proceedings of Robotics: Science and Systems (RSS), 2023.

[37] R.-Z. Qiu, Y. Hu, Y. Song, G. Yang, Y. Fu, J. Ye, J. Mu, R. Yang, N. Atanasov, S. Scherer, and X. Wang, "Learning generalizable feature fields for mobile manipulation," in 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2025, pp. 20 952–20 959. DOI: 10.1109/IROS60139.2025.11246780.

[38] A. Zareian, K. Dela Rosa, D. H. Hu, and S.-F. Chang, "Open-vocabulary object detection using captions," in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 14 388–14 397. DOI: 10.1109/CVPR46437.2021.01416.

[39] M. Ghasemi, A. H. Moosavi, and D. Ebrahimi, A comprehensive survey of reinforcement learning: From algorithms to practical challenges, arXiv preprint arXiv:2411.18892, 2025. arXiv: 2411.18892 [cs.AI]. [Online]. Available: https://arxiv.org/abs/2411.18892.

[40] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, A. Clarkson, M. Yan, B. Budge, Y. Yan, X. Pan, J. Yon, Y. Zou, K. Leon, N. Carter, J. Briales, T. Gillingham, E. Mueggler, L. Pesqueira, M. Savva, D. Batra, H. M. Strasdat, R. De Nardi, M. Goesele, S. Lovegrove, and R. Newcombe, The replica dataset: A digital replica of indoor spaces, 2019. DOI: 10.48550/arXiv.1906.05797. arXiv: 1906.05797 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1906.05797.

[41] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra, "Habitat: A platform for embodied ai research," in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 9338–9346. DOI: 10.1109/ICCV.2019.00943.

[42] A. Wang, L. Liu, H. Chen, Z. Lin, J. Han, and G. Ding. "Yoloe: Real-time seeing anything." arXiv: 2503.07465 [cs.CV]. (2025), [Online]. Available: https://arxiv.org/abs/2503.07465 (visited on 01/02/2026).

[43] P. S. Thomas and E. Brunskill. "Policy gradient methods for reinforcement learning with function approximation and action-dependent baselines." arXiv: 1706.06643 [cs.AI]. (2017), [Online]. Available: https://arxiv.org/abs/1706.06643 (visited on 01/02/2026).

[44] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese, "Gibson env: Real-world perception for embodied agents," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Jun. 2018.

[45] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in

2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 7464–7475. DOI: 10.1109/CVPR52729.2023.00721.

[46] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niebner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3d: Learning from rgb-d data in indoor environments," in 2017 International Conference on 3D Vision (3DV), 2017, pp. 667–676. DOI: 10.1109/3DV.2017.00081.

[47] J. Jiang, Y. Zhu, Z. Wu, and J. Song, "Dualmap: Online open-vocabulary semantic mapping for natural language navigation in dynamic changing scenes," IEEE Robotics and Automation Letters, vol. 10, no. 12, pp. 12 612–12 619, 2025. DOI: 10.1109/LRA.2025.3621942.

[48] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, "Yolo-world: Real-time open-vocabulary object detection," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 16 901–16 911.

[49] R. Kabir, Y. Watanobe, M. R. Islam, and K. Naruse, "Enhanced robot motion block of a-star algorithm for robotic path planning," Sensors, vol. 24, no. 5, p. 1422, 2024. DOI: 10.3390/s24051422.

[50] M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, C. Schuhmann, L. Schmidt, and J. Jitsev, "Reproducible scaling laws for contrastive language-image learning," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 2818–2829.

[51] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik, "Lerf: Language embedded radiance fields," in Proceedings of the IEEE/CVF International Conference on Computer Vision, Oct. 2023, pp. 19 729–19 739.

[52] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, DINOv2: Learning robust visual features without supervision, 2024. DOI: 10.48550/arXiv.2304.07193. arXiv: 2304.07193. [Online]. Available: https://arxiv.org/abs/2304.07193.

[53] X. Zou, J. Yang, H. Zhang, F. Li, L. Li, J. Wang, L. Wang, J. Gao, and Y. J. Lee, "Segment everything everywhere all at once," in Proceedings of the International Conference on Neural Information Processing Systems, Red Hook, NY, USA: Curran Associates, 2023.

[54] S. Schwaiger, S. Thalhammer, W. Wöber, and G. Steinbauer-Wagner, OTAS: Open-vocabulary token alignment for outdoor segmentation, 2025. DOI:

10.48550/arXiv.2507.08851. arXiv: 2507.08851 [cs.RO]. [Online]. Available: https://arxiv.org/abs/2507.08851.

[55] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer, SAM 2: Segment anything in images and videos, 2024. DOI: 10.48550/arXiv.2408.00714. arXiv: 2408.00714 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2408.00714.

[56] D. Maggio, Y. Chang, N. Hughes, M. Trang, D. Griffith, C. Dougherty, E. Cristofalo, L. Schmid, and L. Carlone, "Clio: Real-time task-driven open-set 3d scene graphs," IEEE Robotics and Automation Letters, vol. 9, no. 10, pp. 8921–8928, 2024. DOI: 10.1109/LRA.2024.3451395.

[57] C. Westphal, S. Hailes, and M. Musolesi. "A Generalized Information Bottleneck Theory of Deep Learning." arXiv: 2509.26327 [cs]. (Oct. 14, 2025), [Online]. Available: http://arxiv.org/abs/2509.26327 (visited on 01/10/2026), pre-published.

[58] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in Proceedings of the European Conference on Computer Vision (ECCV), 2020, pp. 405–421.

[59] D. F. Crouse, "On implementing 2D rectangular assignment algorithms," IEEE Transactions on Aerospace and Electronic Systems, vol. 52, no. 4, pp. 1679–1696, 2016. DOI: 10.1109/TAES.2016.140952.

[60] M. Ranzinger, G. Heinrich, J. Kautz, and P. Molchanov, "Am-radio: Agglomerative vision foundation model reduce all domains into one," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 12 490–12 500.

[61] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, "Sigmoid loss for language image pre-training," in 2023 IEEE/CVF International Conference on Computer Vision (ICCV), 2023, pp. 11 941–11 952. DOI: 10.1109/ICCV51070.2023.01100.

[62] B. Yamauchi, "A frontier-based approach for autonomous exploration," in Proceedings of the IEEE International Symposium, Computational Intelligence in Robotics and Automation, Jul. 1997, pp. 146–151. DOI: 10.1109/CIRA.1997.613851. [Online]. Available: https://ieeexplore.ieee.org/document/613851 (visited on 01/14/2026).

[63] S. Macenski and I. Jambrecic, "SLAM toolbox: SLAM for the dynamic world," Journal of Open Source Software, vol. 6, p. 2783, 2021. DOI: 10.21105/joss.02783.

[64] G. Sharir, A. Noy, and L. Zelnik-Manor, An image is worth 16x16 words, what is a video worth? 2021. DOI: 10.48550/arXiv.2103.13915. arXiv: 2103.13915 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2103.13915.

[65] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Niessner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[66] A. Hacinecipoglu, E. Konukseven, and A. Koku, "Pose invariant people detection in point clouds for mobile robots," International Journal of Mechanical Engineering and Robotics Research, vol. 9, no. 5, pp. 709–715, 2020. DOI: 10.18178/ijmerr.9.5.709-715.

[67] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3D Point cloud based object maps for household environments," Robotics and Autonomous Systems, Semantic Knowledge in Robotics, vol. 56, no. 11, pp. 927–941, Nov. 30, 2008, ISSN: 0921-8890. DOI: 10.1016/j.robot.2008.08.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889008001140 (visited on 01/17/2026).

[68] F. Hampel, E. Ronchetti, P. Rousseeuw, and W. Stahel, Robust Statistics: The Approach Based on Influence Functions. 1986, ISBN: 9780471735779. DOI: 10.1002/9781118186435.

[69] M. Minderer, A. Gritsenko, and N. Houlsby, Scaling open-vocabulary object detection, 2024. DOI: 10.48550/arXiv.2306.09683. arXiv: 2306.09683. [Online]. Available: https://arxiv.org/abs/2306.09683.

[70] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D proposal generation and object detection from view aggregation," in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 1–8. DOI: 10.1109/IROS.2018.8594049.

[71] J. Meier, L. Scalerandi, O. Dhaouadi, J. Kaiser, N. Araslanov, and D. Cremers, CARLA drone: Monocular 3d object detection from a different perspective, 2024. DOI: 10.48550/arXiv.2408.11958. arXiv: 2408.11958 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2408.11958.

[72] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[73] A. Gupta, P. Dollár, and R. Girshick, "Lvis: A dataset for large vocabulary instance segmentation," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019.

[74] P. Corke, Robotics, Vision and Control: Fundamental Algorithms in Python (Springer Tracts in Advanced Robotics), 3rd ed. Cham: Springer International Publishing, 2023, vol. 146, ISBN: 978-3-031-06468-5, 978-3-031-06469-2. DOI: 10.1007/978-3-031-06469-2. [Online]. Available: https://link.springer.com/10.1007/978-3-031-06469-2.

[75] Z. Zhang, “A flexible new technique for camera calibration,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 11, pp. 1330–1334, Nov. 2000, ISSN: 1939-3539. DOI: 10.1109/34.888718. [Online]. Available: https://ieeexplore.ieee.org/document/888718/citations (visited on 01/18/2026).

[76] S. Macenski, T. Moore, D. V. Lu, A. Merzlyakov, and M. Ferguson, “From the desks of ros maintainers: A survey of modern & capable mobile robotics algorithms in the robot operating system 2,” Robotics and Autonomous Systems, vol. 168, p. 104 493, 2023, ISSN: 0921-8890. DOI: 10.1016/j.robot.2023.104493. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092188902300132X.

[77] A. Dedieu, G. Zhou, D. George, and M. Lazaro-Gredilla, Learning noisy-OR Bayesian networks with max-product belief propagation, arXiv preprint, 2023. arXiv: 2302.00099 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2302.00099.

[78] M. Colledanchise and L. Natale, “On the implementation of behavior trees in robotics,” IEEE Robotics and Automation Letters, vol. 6, no. 3, pp. 5929–5936, 2021. DOI: 10.1109/LRA.2021.3087442.

[79] S. Pütz, J. Santos Simón, and J. Hertzberg, “Move base flex a highly flexible navigation framework for mobile robots,” in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 3416–3421. DOI: 10.1109/IROS.2018.8593829.

[80] Z. Zhou, J. Song, X. Xie, Z. Shu, L. Ma, D. Liu, J. Yin, and S. See, “Towards building AI-CPS with NVIDIA Isaac Sim: An industrial benchmark and case study for robotics manipulation,” in Proceedings of the IEEE–ACM International Conference on Software Engineering, Software Engineering in Practice, 2024, pp. 263–274. DOI: 10.1145/3639477.3639740.

[81] S. Salimpour, J. Peña-Queralta, D. Paez-Granados, J. Heikkonen, and T. Westerlund. “Sim-to-real transfer for mobile robots with reinforcement learning: From NVIDIA isaac sim to gazebo and real ROS 2 robots.” arXiv: 2501.02902 [cs.RO]. (2025), [Online]. Available: https://arxiv.org/abs/2501.02902 (visited on 01/14/2026).

[82] S. Lee, T. Kim, J. Chae, and K.-J. Park, Optimizing ROS 2 communication for wireless robotic systems, 2025. DOI: 10.48550/arXiv.2508.11366. arXiv: 2508.11366 [cs.NI]. [Online]. Available: https://arxiv.org/abs/2508.11366.

[83] K. Zakka, B. Tabanpour, Q. Liao, M. Haiderbhai, S. Holt, J. Y. Luo, A. Allshire, E. Frey, K. Sreenath, L. A. Kahrs, C. Sferrazza, Y. Tassa, and P. Abbeel. “MuJoCo Playground.” arXiv: 2502.08844 [cs]. (Feb. 12, 2025), [Online]. Available: http://arxiv.org/abs/2502.08844 (visited on 01/19/2026), pre-published.

[84] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,”

in 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), vol. 3, 2004, pp. 2149–2154. DOI: 10.1109/IROS.2004.1389727.

[85] X. Puig, E. Undersander, A. Szot, M. D. Cote, T.-Y. Yang, R. Partsey, R. Desai, A. W. Clegg, M. Hlavac, S. Y. Min, V. Vondruš, T. Gervet, V.-P. Berges, J. M. Turner, O. Maksymets, Z. Kira, M. Kalakrishnan, J. Malik, D. S. Chaplot, U. Jain, D. Batra, A. Rai, and R. Mottaghi, Habitat 3.0: A co-habitat for humans, avatars and robots, 2023. DOI: 10.48550/arXiv.2310.13724. arXiv: 2310.13724 [cs.HC]. [Online]. Available: https://arxiv.org/abs/2310.13724.

[86] S. K. Ramakrishnan, A. Gokaslan, E. Wijmans, O. Maksymets, A. Clegg, J. Turner, E. Undersander, W. Galuba, A. Westbury, A. X. Chang, M. Savva, Y. Zhao, and D. Batra, Habitat-matterport 3d dataset (HM3D): 1000 large-scale 3d environments for embodied AI, 2021. DOI: 10.48550/arXiv.2109.08238. arXiv: 2109.08238. [Online]. Available: https://arxiv.org/abs/2109.08238.

[87] S. Macenski, A. Soragna, M. Carroll, and Z. Ge, "Impact of ROS 2 node composition in robotic systems," IEEE Robotics and Automation Letters, vol. 8, no. 7, pp. 3996–4003, 2023. DOI: 10.1109/LRA.2023.3279614.

[88] R. Sapkota and M. Karkee. "Ultralytics YOLO evolution: An overview of YOLO26, YOLO11, YOLOv8 and YOLOv5 object detectors for computer vision and pattern recognition." arXiv: 2510.09653 [cs.CV]. (2025), [Online]. Available: https://arxiv.org/abs/2510.09653 (visited on 01/14/2026).

[89] D. Li, J. Li, H. Le, G. Wang, S. Savarese, and S. C. H. Hoi, LAVIS: A library for language-vision intelligence, 2022. DOI: 10.48550/arXiv.2209.09019. arXiv: 2209.09019 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2209.09019.

List of Figures

List of Tables

A Behavior Tree Details

The behavior tree (exported from Groot2) orchestrates the alternation between (i) perception-driven object confirmation and (ii) frontier-based exploration, while maintaining a persistent semantic map across sequential queries.

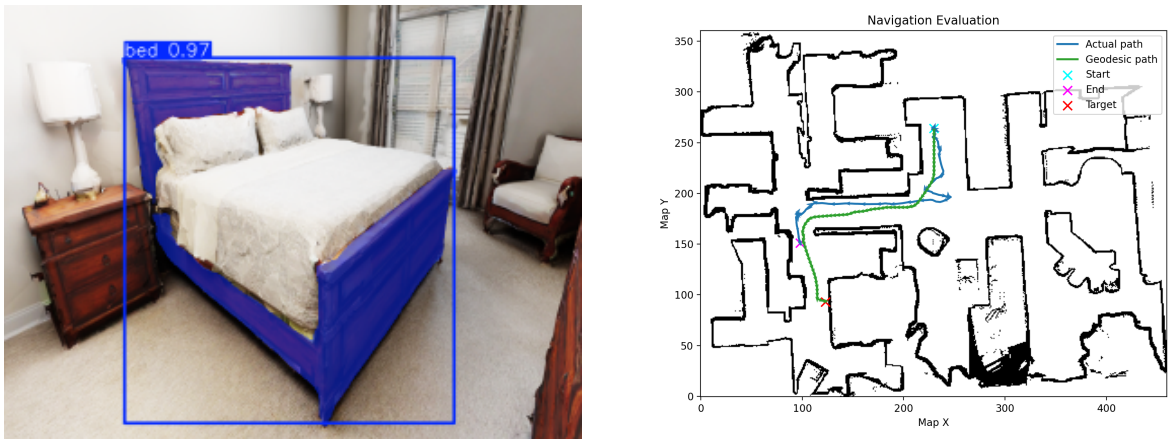
Figure 26: High-level behavior tree of SAGE. Control flow nodes (selectors/sequences) gate between an object-confirmation branch and an exploration branch. Leaf nodes correspond to atomic actions or condition checks.

Table 15 summarizes the role and outcome semantics of the main subtrees and leaf nodes. This reference complements the method description in Chapter 3 by making the execution logic explicit at the decision-making level.

BT Element	Purpose and outcome semantics
Object confirmation branch	Uses the detection graph nodes G_t^{det} to decide whether the queried target is present. The branch returns Success if at least one graph node exceeds the configured confirmation threshold; otherwise, it returns Failure, which triggers exploration. Internally, confirmation is executed in three stages: (a) permissive candidate acquisition with a low detection threshold, (b) navigation and viewpoint alignment toward the highest-scoring detection candidate, and (c) final verification under a high detection threshold.
Exploration branch	Selects a frontier candidate according to the fused frontier utility and navigates to the corresponding viewpoint. The branch returns Success once a new viewpoint is reached; it returns Failure if navigation fails or if no valid frontier remains.
Re-orientation / observation	Performs an observation routine (e.g., in-place rotation or local viewpoint refinement) to update semantic evidence for graph nodes, including value-map scoring based on cosine similarity. The routine returns Success after completing the observation cycle.
Termination condition	Terminates an episode if the target is confirmed or if the episode budget is exhausted (e.g., time, steps, or available frontiers).

Table 15: Summary of the main behavior tree elements used to orchestrate SAGE. The exact set of leaf nodes depends on the task configuration, but the success/failure semantics follow the roles stated above.

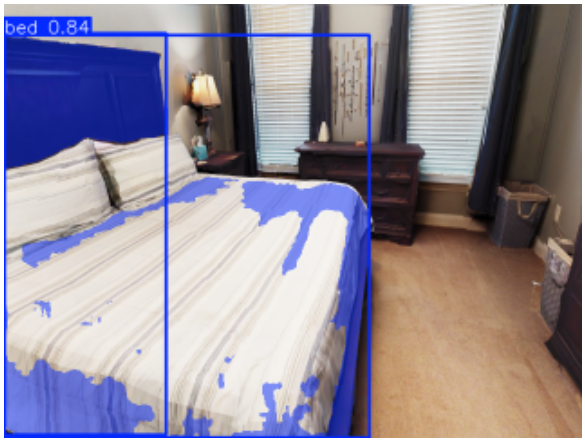
B Additional Experimental Results



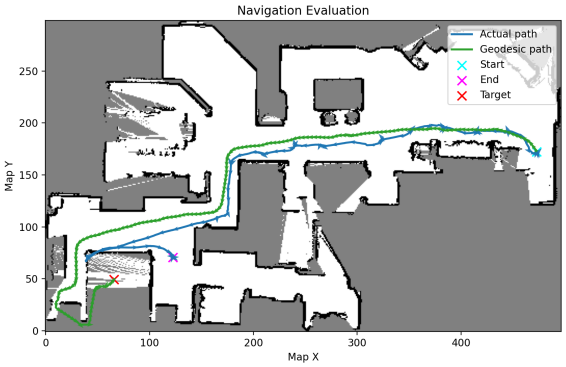
(a) Object detection overlay (bed)

(b) Navigation trajectory

Figure 27: RQ1 example (scene `00848-ziup5kvtCCR`): object detection overlay and executed navigation trajectory for the target object bed.



(a) Object detection overlay (bed)



(b) Navigation trajectory

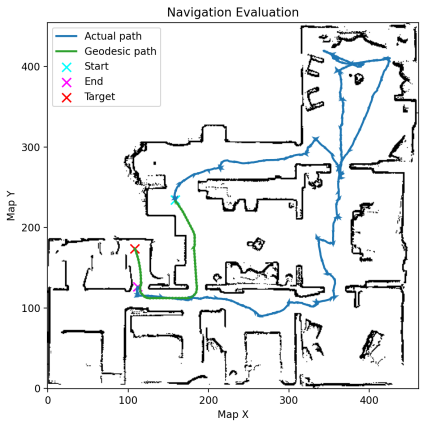
Figure 28: RQ1 example (scene `00800-TEEsavR23oF`): object detection overlay and executed navigation trajectory for the target object bed.

(a) Object detection overlay (toilet)

(b) Navigation trajectory

Figure 29: RQ1 example (scene `00814-p53SfW6mjZe`): object detection overlay and executed navigation trajectory for the target object toilet.

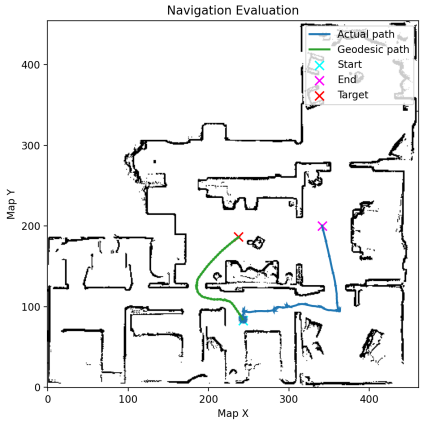
(a) Object detection overlay (toilet)



(b) Navigation trajectory

Figure 30: RQ1 example (scene `00824-Dd4bFSTQ8gi`): object detection overlay and executed navigation trajectory for the target object toilet.

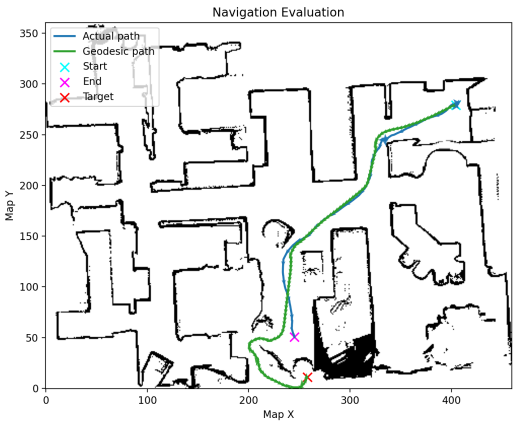
(a) Object detection overlay (plant)



(b) Navigation trajectory

Figure 31: RQ1 example (scene `00824-Dd4bFSTQ8gi`): object detection overlay and executed navigation trajectory for the target object plant.

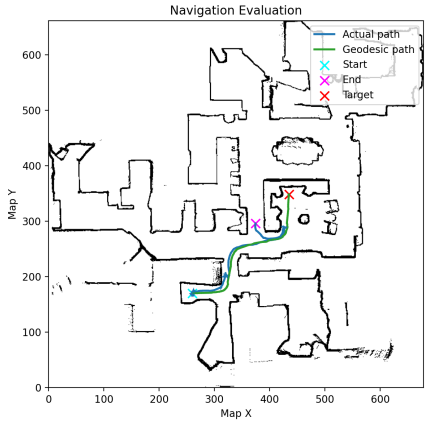
(a) Object detection overlay (christmas tree)



(b) Navigation trajectory

Figure 32: RQ1 example (scene `00848-ziup5kvtCCR`): object detection overlay and executed navigation trajectory for the target object christmas tree.

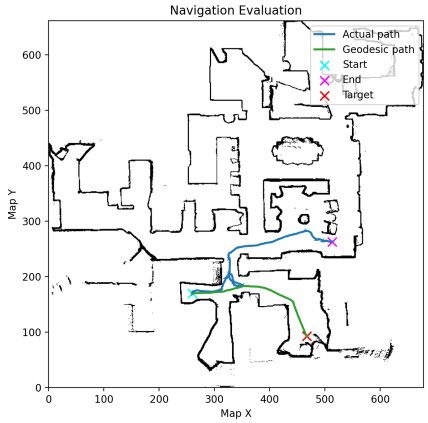
(a) Object detection overlay (sofa)



(b) Navigation trajectory

Figure 33: RQ1 example (scene `00876-mv2HUxq3B53`): object detection overlay and executed navigation trajectory for the target object sofa.

(a) Object detection overlay (chair)



(b) Navigation trajectory

Figure 34: RQ1 failure example (scene `00876-mv2HUxq3B53`): FP detection caused by a visually correct but physically unreachable object. The chair is correctly detected in the image but is located behind a window outside the navigable space, leading the robot to navigate towards the window under the assumption that the object is reachable.

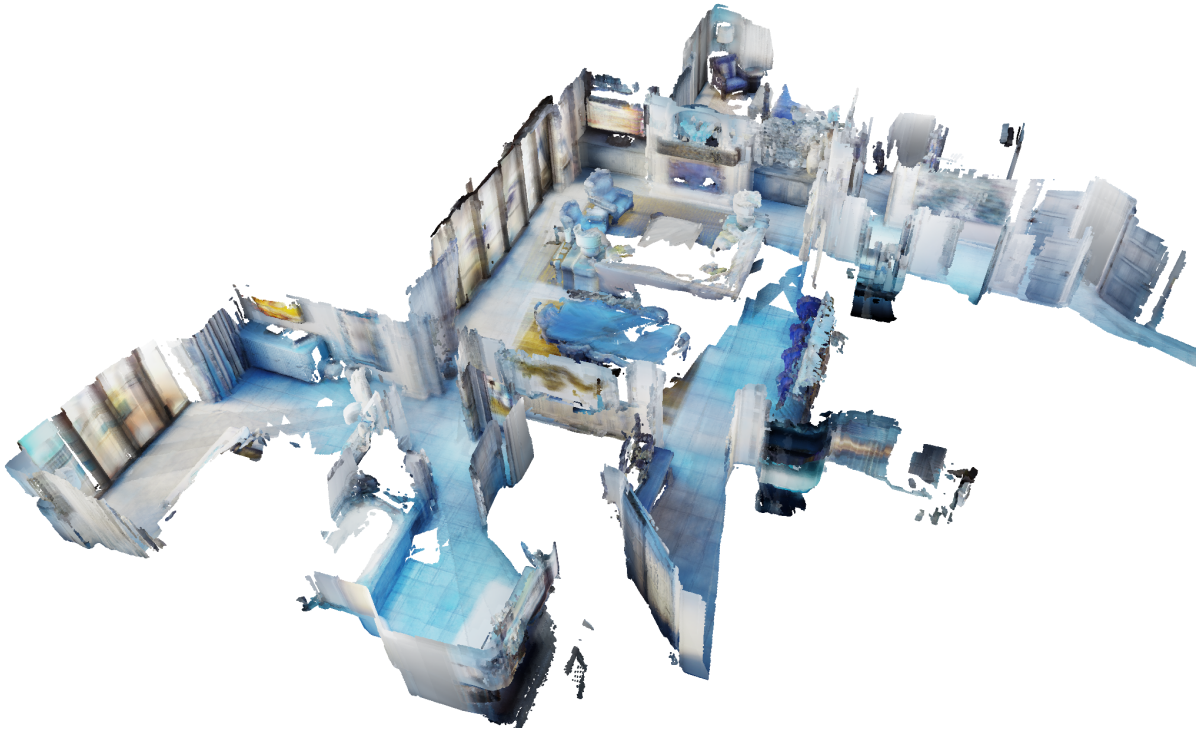


Figure 35: Example semantic map built with a total voxel count of 191,941 using OpenFusion with a maximum voxel limit of 200,000. The map effectively captures the environment's structure and semantics within the specified memory constraints.

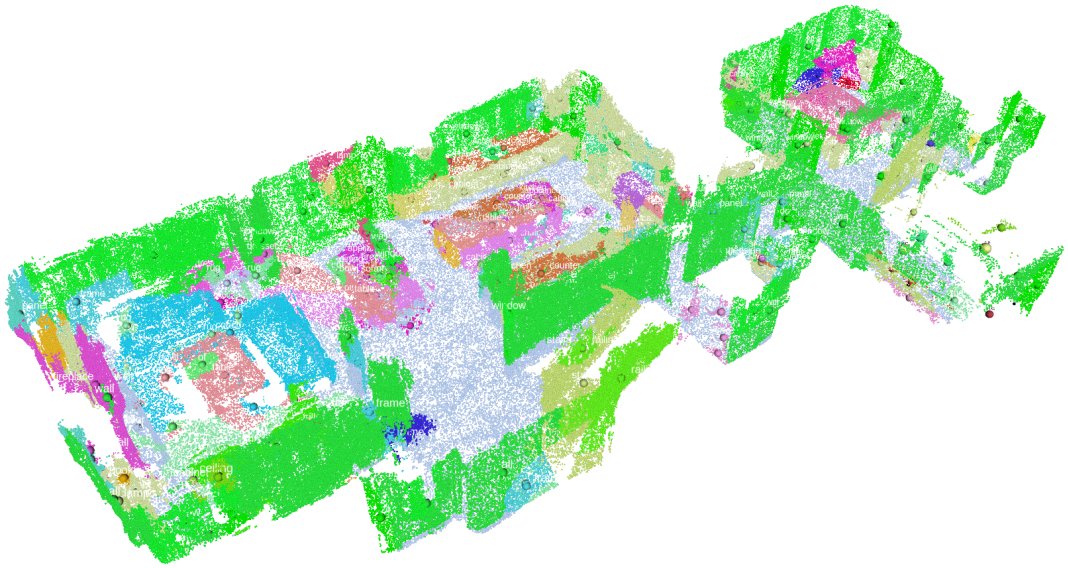


Figure 36: Semantic evaluation point cloud generated with OpenFusion [35]. The point cloud is constructed using class predictions provided by Matterport3D [86] and subsequently filtered by object classes, clustered, and reduced to object centroids. These centroids serve as ground truth object locations for quantitative evaluation.