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

Supervisor: Simon Schwaiger, MSc

Vienna, December 15, 2025

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, December 15, 2025 Signature

Kurzfassung

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Schlagworte: Keyword1, Keyword2, Keyword3, Keyword4

Abstract

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Keywords: Keyword1, Keyword2, Keyword3, Keyword4

Acknowledgements

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Contents

1 Introduction **1**
- 1.1 Problem Statement . . . 1
- 1.2 Scientific Contribution . . . 5
- 1.3 Thesis Structure . . . 6

2 State of the Art **6**
- 2.1 Geometric Exploration . . . 7
- 2.2 Vision-Language-Guided Exploration . . . 8
- 2.3 Object Detection and Promptable Models . . . 14

3 Methods **14**
- 3.1 System Overview . . . 15
- 3.2 Semantic Frontier Exploration . . . 15
- 3.3 Persistent Semantic 3D Mapping . . . 17
- 3.4 Promptable Zero-Shot Detection . . . 18
- 3.5 Fusion Strategy . . . 20
- 3.6 Behavior Tree for Semantic-Guided Exploration . . . 20

4 Implementation **23**
- 4.1 Simulation Environment . . . 23
- 4.2 Dataset . . . 23
- 4.3 Used Software . . . 23
- 4.4 Used Hardware . . . 24
- 4.5 Evaluation Metrics . . . 24

5 Discussion and Results **30**
- 5.1 Experiment 1: Benchmarking on Matterport Scenes . . . 31
- 5.2 Experiment 2: Impact of Exploration–Memory Weighting . . . 31
- 5.3 Experiment 3: Sensitivity to Semantic Map Granularity . . . 31
- 5.4 Experiment 4: Effect of Multi-Source Semantic Fusion . . . 31
- 5.5 Experiment 5: System Efficiency and Real-World Validation . . . 31

6 Summary and Outlook **31**

Bibliography **32**

List of Figures **35**

List of Tables **36**

List of source codes **37**

A Appendix A **38**

B Appendix B **39**

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) and Vision-Language Models (VLMs) have enabled robots to interpret and act upon complex, open-ended instructions expressed in natural language. These developments have catalyzed a paradigm shift toward semantic reasoning and zero-shot generalization, allowing robots to understand 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. In service robotics, for instance, 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) [3]. Similarly, in Search and Rescue (SAR) operations, 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 becoming increasingly essential. Robots must be capable of interpreting abstract human instructions, reasoning about both spatial and semantic context, and conducting efficient searches in dynamic, partially known environments. These challenges motivate the development of a unified framework that bridges geometric exploration with semantic scene understanding, enabling autonomous agents to perform open-vocabulary, goal-directed exploration guided by high-level semantic input.

1.1 Problem Statement

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. Such methods, including those based on occupancy grids or point cloud representations, are particularly effective for coverage and mapping tasks and have been successfully extended to multi-robot systems for large-scale exploration. 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.

To address this limitation, recent research has focused on integrating semantic perception into exploration frameworks. 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 reinforcement learning-based policies to guide robots toward regions that are semantically relevant to a given target description.

Approach	**Training Required**	**Real-Time**	**Semantic Reasoning Model**
VLFM [4]	✗ (zero-shot)	✓	BLIP-2 + GroundingDINO + SAM
SemUtil [5]	✗ (training-free)	✓	Mask R-CNN + CLIP + BERT
ESC [6]	✗ (zero-shot)	✓	GLIP + DeBERTa / ChatGPT reasoning
LGX [7]	✗ (zero-shot)	✓	GPT-3 + GLIP + BLIP
CoW [8]	✗ (zero-shot)	✓	CLIP similarity scoring
ZSON [9]	✓ (RL pretraining)	✗	CLIP-based RL policy
PONI [10]	✓ (supervised)	✗	Learned potential-field network
PIRLNav [11]	✓ (BC + RL)	✗	DINO-based CNN-RNN policy

Table 1: Overview of semantic zero-shot and trained exploration approaches.

Approaches such as ZSON [9], PONI [10], and PIRLNav [11] 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 novel environments or object categories not encountered during training. In contrast, zero-shot methods such as VLFM [4], SemUtil [5], ESC [6], LGX [7], and CoW [8] 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 [6] and LGX [7], 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 [4] achieves real-time performance, it relies on multiple computationally heavy models (GroundingDINO, SAM, and BLIP-2) that require high-end GPUs with up to 16 GB of VRAM, which is impractical for embedded robotic systems.

Overall, these 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 search tasks. A persistent global memory would enable robots to exploit past observations, recall detected objects, and avoid revisiting known regions, thereby improving navigation efficiency, scalability, and robustness in open-vocabulary, real-world environments. 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 [12]	✗ (zero-shot)	✓	2D probabilistic feature field	✓ (frontier-based)
ConceptGraphs [13]	✗ (pretrained models)	✗	3D scene graph	✓ (LLM-planner)
SemExp [14]	✓ (RL + supervised)	✗	2D semantic occupancy map	✓ (learned policy)
GeFF [15]	✓ (ScanNet pretrain)	✓	Implicit 3D feature field	✗ (passive)
RayFronts [16]	✗ (foundation model)	✓	Hybrid voxel + ray field	✗ (planner-agnostic)
VLMaps [17]	✗ (pretrained LSeg/CLIP)	✗	2.5D open-vocab grid	✓ (frontier-compatible)
Pigeon [18]	✓ (RLVR fine-tune)	✓	Point-of-Interest snapshot memory	✓ (reasoning-aware)

Table 2: Overview of persistent or memory-based semantic mapping approaches.

Similar to the previously discussed category, methods such as SemExp [14], Pigeon [18], and GeFF [15] rely on offline training to develop navigation policies that utilize persistent semantic representations. This dependence on extensive training limits their adaptability to novel environments and unseen object categories. Other approaches, such as ConceptGraphs [13] and VLMaps [17], 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 OneMap [12] achieves real-time performance on embedded hardware such as the Jetson Orin AGX, its computational cost limits operation to approximately 2 Hz, which may be insufficient for high-speed navigation tasks. Additionally, noise in depth perception directly degrades the quality of the probabilistic feature map, reducing overall semantic reliability. The detector used in OneMap relies solely on VLM-based CLIP image–text similarity, which makes it prone to false positives under open-vocabulary conditions, a phenomenon commonly referred to as *open-vocabulary drift*.

GeFF [15] 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 ar-

bitrary environments. RayFronts [16] 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 [4], CoW [8], LGX [7], ESC [6]	No long-term fusion or recall; repeated exploration of known areas.
Offline training required	ZSON [9], PONI [10], PIRLNav [11], SemExp [14]	Heavy RL/supervised training; poor adaptability to new scenes.
No balance between exploration and memory	OneMap [12], RayFronts [16], VLMaps [17]	Either passive mapping or short-term exploration; inefficient search.
No zero-shot exploration	VLFM [4], CoW [8], LGX [7]	Detect novel objects but fail to explore unseen regions strategically.
Premapping needed	ConceptGraphs [13], VLMaps [17], GeFF [15]	Depend on pre-recorded data; not suited for online autonomy.
Limited robustness	PONI [10], SemExp [14], PIRLNav [11]	Closed-set categories; fragile under real-world variation.
Low real-world applicability	ConceptGraphs [13], VLMaps [17], Pigeon [18]	High GPU cost or simulation-only evaluation; limited deployability on mobile robots.

Table 3: Identified gaps in existing semantic exploration frameworks.

These observations reveal a clear research gap: there is a need for a framework that enables autonomous robots to explore unknown environments in a zero-shot manner while maintaining a persistent, incrementally updated 3D semantic map. Such a system should support continuous updates, querying, and reasoning over spatial and semantic information without the need for retraining or pre-mapping.

By leveraging pretrained VLMs, the dependency on computationally expensive Deep Reinforcement Learnings (DRLs) or supervised training pipelines can be eliminated, enabling flexible object and scene understanding beyond fixed category sets. Furthermore, a hybrid exploration strategy is essential, allowing the robot to dynamically balance between discovering new regions (*exploration*) and exploiting previously stored knowledge (*memory*) depending on environmental stability and task objectives. This trade-off is particularly relevant in dynamic settings, where semantic frontiers evolve over time, as well as in static domains, where persistent knowledge can be efficiently reused.

For real-world deployment, the proposed framework must prioritize computational efficiency

and robustness to sensor noise and environmental variability. Even under constrained hardware resources, it should sustain real-time operation on embedded platforms without compromising latency or inference stability.

Finally, unlike prior works such as OneMap [12] and VLFM [4], which rely on single-source detection pipelines, the envisioned framework integrates a multi-source fusion strategy. By combining detection confidence, semantic similarity, and memory-based reliability, the system aims to enhance detection robustness, suppress false positives, and achieve consistent open-vocabulary reasoning during long-term semantic exploration.

1.2 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 search guided by open-vocabulary text queries. The system 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 world model. Adaptive 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 framework further investigates 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 work 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 system, 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?**
   Metrics: Performance is quantified in terms of 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 [12], VLFM [4], and Pigeon [18].

2. **How does the interaction between live exploration and accumulated semantic memory influence overall system performance?**
   Metrics: 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 does the granularity of semantic map retrieval affect map quality, and can dynamic weighting between exploration and memory compensate for potential noise?**
   Metrics: The semantic granularity in the 3D semantic mapper is varied while adjusting exploration weight to evaluate effects on 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?**
   Metrics: Precision, Recall, F1-Score, Confusion Matrix, and SR under different fusion weight configurations across COCO, open-vocabulary, and zero-shot classes.

5. **What is the computational footprint and real-world robustness of the hybrid framework?**
   Metrics: Frames Per Second (FPS), GPU/CPU usage, inference latency, and detection stability under sensor noise during physical deployment on a mobile robot.

These research questions guide the design of the experimental evaluation, where each question is systematically addressed through targeted ablation studies, comparative benchmarks, and real-world validation presented in Chapter 5.

1.3 Thesis Structure

This work is structured as follows: Chapter 2 describes the state-of-the-art in semantic exploration, persistent mapping, and object detection. 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

In this chapter, the current state of research in semantic multi-object search, map reconstruction, and object detection is reviewed. The goal is to identify strengths and limitations of existing methods and establish the technological context for the proposed hybrid approach. The chapter is divided into three key areas: approaches for searching multiple objects semantically, techniques for building and maintaining persistent semantic maps, and recent advances in object detection and promptable models for open-vocabulary tasks.

2.1 Geometric Exploration

A *frontier* is defined as the boundary between known free space and unknown regions of the environment [19]. 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. [21] 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 [19].

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 Vision-Language-Guided 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.

Recent research has introduced DRL and supervised learning approaches to train agents capable of performing semantic object search [9–11]. These methods differ in architecture and learning paradigm but share a common goal of integrating visual understanding with navigation policies.

Majumdar et al. [9] proposed Zero-Shot Object Navigation (ZSON), a zero-shot object navigation framework that leverages the multimodal embedding space of CLIP [22] to guide exploration policies trained via reinforcement learning. During training, the agent's observation space consists of previous actions and RGB images, while the navigation goal is represented by the CLIP embedding of an image containing the target object. The reward function is defined as:

$$r_t = r_{\text{success}} + r_{\text{angle-success}} - \Delta d_{tg} - \Delta a_{tg} + r_{\text{slack}}, \tag{1}$$

where r_{success} provides a large positive reward for successfully locating the target object, $r_{\text{angle-success}}$ rewards correct orientation toward the object, Δd_{tg} penalizes increased distance to the target, Δa_{tg} penalizes angular deviation, and r_{slack} applies a small negative reward to encourage efficiency. At inference time, the image-based embedding is replaced by the CLIP embedding of a text description of the target object, enabling zero-shot generalization to unseen objects. The agent's action space includes four discrete actions: *move forward*, *turn left*, *turn right*, and *stop*.

While ZSON demonstrates strong zero-shot generalization and effective object localization within trained domains, it lacks explicit mechanisms for structured exploration in unknown regions. Since the learned policy primarily directs the agent toward locations visually similar to previously encountered objects, it may become trapped in familiar areas and fail to discover new regions. Moreover, the reliance on DRL training reduces interpretability and generalizability; retraining is often required when visual conditions, environment layouts, or sensor modalities differ significantly from the training distribution.

Building on this line of research, Ramrakhya et al. [11] introduced Pretraining with Imitation and RL Finetuning (PIRLNav), a two-stage pretraining framework for vision-language navigation that combines Behavioral Cloning (BC) and Reinforcement Learning (RL) to learn generalizable exploration policies. The approach first leverages large-scale human demonstrations to

imitate expert navigation behavior and subsequently fine-tunes the policy using RL to optimize long-term performance.

In the first stage, the policy network is pretrained via BC on a dataset containing approximately 77,000 human expert demonstrations. The objective is to minimize the negative log-likelihood of actions given the corresponding observations, as defined by:

$$\theta^* = \arg\min_{\theta} \sum_{(o_t, a_t)} -\log \pi_\theta(a_t \mid o_t), \tag{2}$$

where θ denotes the parameters of the policy network, and (o_t, a_t) are observation-action pairs derived from expert trajectories. The observation space o_t includes RGB images, GPS and compass pose information (HM3D coordinates), a one-hot encoded target object category, and an implicit temporal memory implemented via a gated recurrent unit (GRU). The action space is discrete and consists of *move forward*, *turn left*, *turn right*, and *stop*. This pretraining phase enables the agent to acquire human-like navigation behavior, producing a policy that maps high-dimensional sensory inputs to navigation commands.

In the second stage, the pretrained policy is fine-tuned using DRL to maximize long-term rewards associated with successful object-goal navigation. The actor network is initialized with the weights obtained from the BC stage, while the critic (value function) is first trained independently to stabilize learning before both networks are optimized jointly using Proximal Policy Optimization (PPO). The overall objective is to maximize the expected cumulative discounted reward over trajectories sampled from the policy:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=1}^{T} \gamma^{t-1} r_t \right], \tag{3}$$

where τ represents trajectories of observations, actions, and rewards generated under policy π, r_t is the scalar reward at time step t, and γ is the discount factor. A reward of $r_t = 1$ is assigned when the agent stops within a defined range of the target and the object is within its Field of View (FOV). Notably, PIRLNav does not employ an explicit object detector; instead, object recognition is implicitly encoded in the policy through visual representations learned from pretrained Self-Distillation with No Labels (DINO) [23] features during pretraining.

While PIRLNav achieves strong performance in simulated benchmarks, it inherits several limitations typical of end-to-end reinforcement learning systems. The policy exhibits limited interpretability and lacks explicit semantic memory, preventing long-term reasoning across multiple goals. Furthermore, retraining is required when the visual domain or sensor configuration changes significantly, and generalization strongly depends on the similarity between training and deployment environments.

To overcome the black-box nature of end-to-end policies, Ramakrishnan et al. [10] proposed Potential Functions for ObjectGoal Navigation with Interaction-free Learning (PONI), a map-based object-goal navigation framework that decouples high-level exploration decisions from low-level navigation control. PONI learns a potential-field network that operates on a partial allocentric semantic map, where free space, occupied space, unknown regions, and detected object categories are encoded in distinct semantic channels. The network predicts two dense

potential fields aligned with the current map: an *area potential* U_t^a, which promotes the exploration of large unexplored regions, and an *object potential* U_t^o, which estimates the likelihood of proximity to the target object.

The potential-field network is trained in a fully supervised, interaction-free manner using annotated semantic maps from simulated environments. During training, ground-truth area and object potentials are computed offline by exploiting complete knowledge of free space and object locations. The model is optimized to regress these potentials using a pixel-wise loss applied to frontier cells. At inference time, the network predicts both potentials from the current partial map without access to ground-truth information. Geometric frontiers are extracted from the occupancy map and scored by sampling the predicted potentials according to:

$$U_t(f) = \alpha\, U_t^a(f) + (1-\alpha)\, U_t^o(f), \tag{4}$$

where α controls the trade-off between exploration and exploitation. The frontier with the highest score is selected as the next navigation goal, which is then reached using a deterministic path planner rather than a learned action policy.

While PONI achieves efficient and interpretable exploration behavior, it depends on category-level semantic inputs and assumes a fixed, closed set of object classes encountered during training. Consequently, it cannot generalize to unseen categories or open-vocabulary settings, and its reliance on dense semantic annotations limits applicability in real-world scenarios where such data are unavailable or noisy [10].

Overall, these RL-based and supervised learning approaches illustrate the potential of combining semantic understanding with learned navigation policies but reveal persistent shortcomings that motivate the shift toward pretrained vision-language models. Key limitations include:

- **Extensive training requirements:** Large datasets and prolonged training are needed, reducing adaptability to new environments.
- **High computational demands:** Complex architectures hinder real-time operation on embedded systems.
- **Closed-set semantics:** Models typically rely on a fixed set of object categories, limiting open-vocabulary reasoning.
- **Lack of persistent memory:** No long-term spatial or semantic memory is maintained, causing redundant exploration.
- **Limited interpretability:** End-to-end learning obscures decision-making and complicates integration with classical navigation frameworks.

Therefore, recent research has shifted toward leveraging large-scale pretrained vision-language models (VLMs) that provide rich semantic representations and zero-shot generalization capabilities. By integrating these models into modular exploration frameworks, it becomes possible to overcome many of the limitations associated with learned policies while retaining the benefits of semantic reasoning for efficient multi-object search.

A representative example of this paradigm is the Exploration with Semantic Cues (ESC) framework proposed by Zhou et al. [6], which augments traditional frontier-based exploration with semantic cues derived from pretrained VLMs. Specifically, ESC combines a grounded object detector, Grounded Language-Image Pretraining (GLIP) [24], 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, as formulated in Equation 5:

$$P(F) = P(F \mid d_i^t, o^t, r^t), \tag{5}$$

where $P(F)$ denotes the probability of selecting frontier F, conditioned on detected objects d_i^t, current 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.

In practice, ESC employs GLIP as the detection backbone to compute 2D bounding boxes, class labels, and confidence scores for objects within the robot's FOV. While effective in simulation, the approach introduces notable computational overhead due to repeated LLM inference and PSL optimization. Moreover, its performance is highly dependent on the reliability of object detections-false positives can misguide frontier scoring, and incorrect commonsense priors may further bias exploration.

Through ablation studies, Zhou et al. 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 reasoning-heavy systems, Gadre et al. [8] proposed Clip on Wheels (CoW), a lightweight vision-language exploration framework that relies purely on image-text alignment from CLIP [22] without requiring explicit frontier detection, semantic mapping, or object segmentation. The method guides the robot toward directions with the highest cosine similarity between the current visual observation and the target object description, defined as:

$$\cos\_\mathrm{sim}(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \, \|\mathbf{v}\|}, \tag{6}$$

where $\mathbf{u}$ and $\mathbf{v}$ denote the CLIP embeddings of the current image and the target text prompt, respectively. The robot moves forward when the similarity exceeds a predefined threshold and rotates otherwise to maximize alignment with the target.

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. [5] introduced Semantic Utility Maps for Object Goal Navigation (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 [5].

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) [5].

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.

Unlike [5], which converts classes of detection into a value map, and possibly loosing information or adding misinformation,VLFM [4] directly uses the image-text similarity to compute the from the entire image. The cosine similarity 6 [8] is computed with a pre-trained VLM from BLIP-2 [25] between the current observation and the target text query. The similarity values are then projected onto a 2D occupancy grid in a FOV mask, where towards the border of the image FOV mask values are lower due to lower confidence of the detection (gaussian weighting). The resulting value map is then masked with a geometric frontier map to select the next exploration goal. VLFM [4] uses GroundingDINO [26], for zero-shot detection or YOLOv7 for COCO [3] trained classes, as object detectors to detect objects in the current observation, which are then segmented with SAM [27] to get precise object masks. The detected objects are then used to determine if the target object is in the current observation and stop the exploration and navi-

gate towards the object. The results are promising, however, VLFM [4] also suffers from false positives in the zero-shot detection setting, which can lead to wrong stopping decisions, due to only either having a single source of detection pipeline (GroundingDINO [26] or YOLOv7). The value map is episodic and not persistent over time, which can lead to revisiting already explored areas, which is inefficient for multi-object search. VLFM has real-time properties, but due to using 4 different models (BLIP-2 [25], GroundingDINO [26]/YOLOv7, SAM [27]), it uses 16 GB VRAM on an NVIDIA RTX 4090 GPU, which is quite high for real-time applications on mobile robots.

Semantic Scene Reconstruction

- Overview of approaches to build and update semantic maps during exploration:
  - 2D grid maps
  - Pointclouds
  - Voxel grids
  - Octomaps
  - Scene graphs
  - Neural Radiance Fields (NeRFs)
  - Feature Fields
- Techniques for fusing sensor data into persistent 2D/3D representations:
  - Storing Visual Embeddings (e.g., CLIP features) in 3D maps for semantic querying.
  - Incremental updating of semantic labels based on new observations.
  - Handling uncertainty and conflicting detections over time.
- Comparison of representations (Octomaps, point clouds, voxel grids) in terms of:
  - Memory efficiency.
  - Ability to store semantic labels persistently.
- Discussion of . . .
  - ConceptGraphs
  - ConceptGraph-Online
  - OpenFusion
  - Clio
  - OpenScene
  - GeFF
  - CLIP-Fields
  - ConceptFusion

  - VLMaps
  - LERF

  as examples of global 3D semantic maps.
- Limitations in updating or correcting the map after wrong detections.

Persistent Semantic Mapping for Exploration

2.3 Object Detection and Promptable Models

- Review of traditional and open-vocabulary object detection methods.
- Analysis of grounding-capable detectors and segmentation models for zero-shot tasks.
- Specific evaluation of the following models for their suitability in semantic multi-object search:
  - YOLO-E
  - GroundingDINO
  - MobileSAM
  - GroundedSAM
  - SEEM
  - OWL-ViT
  - MaskDINO
  - GLIP
- Discussion of promptable vision-language models supporting multi-modal queries (text, image, audio).
- Challenges with false positives in zero-shot settings and their implications for reliable multi-object detection.

3 Methods

This chapter details the methods developed for semantic exploration, persistent 3D mapping, promptable object detection, and robust fusion strategies for multi-object search.

3.1 System Overview

- Presentation of the overall architecture of the exploration (3.2), detection (3.4), mapping (3.3), and fusion (3.5) pipeline.
- Description of data flow between exploration (frontier evaluation), detection (promptable models), and memory (persistent semantic mapping), as shown in Figure 8.
- Explanation of how exploration and mapping components interact to progressively build a semantic understanding of the environment.

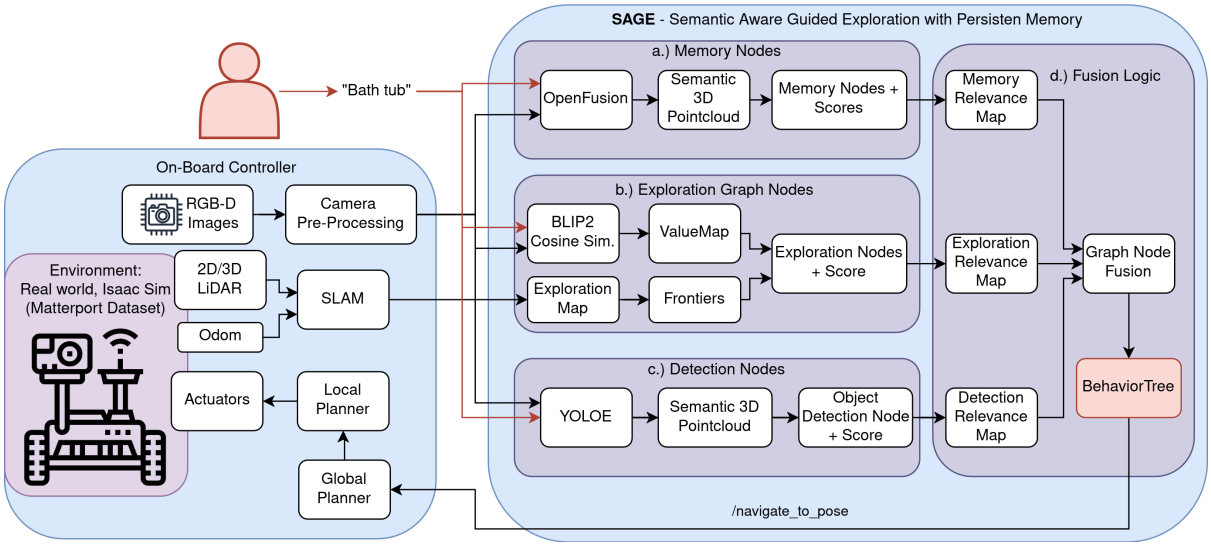

Figure 1: System architecture

3.2 Semantic Frontier Exploration

Exploration 2D Occupancy Map

- The SLAM map is used for navigation.
- Generating frontiers for each prompt would require deleting and rebuilding the SLAM map.
- This approach is inefficient and impractical for navigation.
- Therefore, a separate 2D occupancy grid is created exclusively for exploration.
- The exploration map is constructed by:
  - collecting robot poses, and
  - performing raytracing against the parent SLAM map.
- For each new semantic prompt:

- all stored poses are cleared, and
- the exploration occupancy grid is rebuilt from scratch.

Frontier Detection and Calculation

- Detection of frontiers on a 2D occupancy grid to identify candidate regions for exploration.
- Application of classical frontier-based exploration algorithms extended with semantic information.

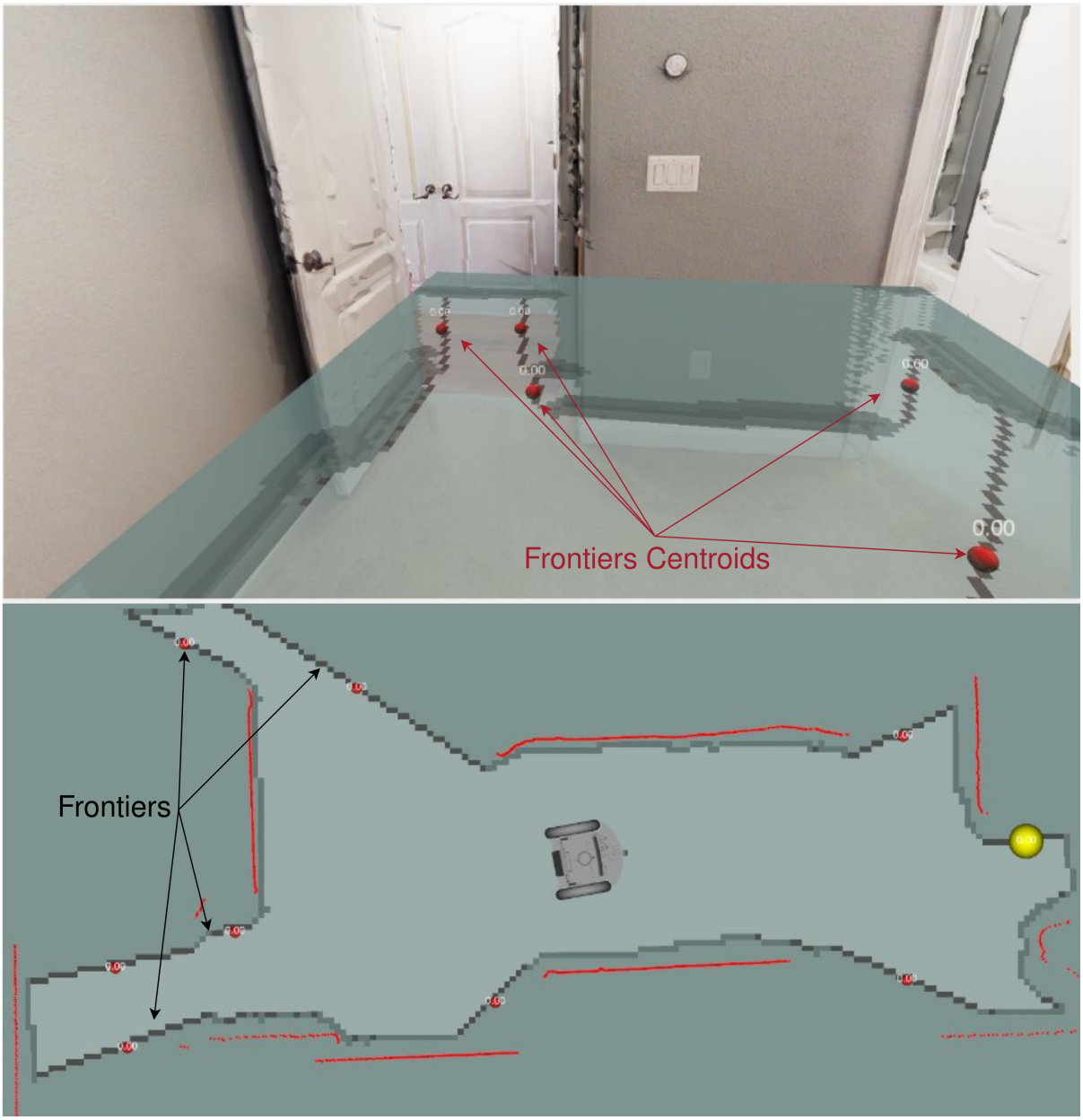

Figure 2: Frontier detection on occupancy grid

Value Map Generation using Vision-Language Models

- Computation of value maps by evaluating cosine similarity between text queries and scene observations.

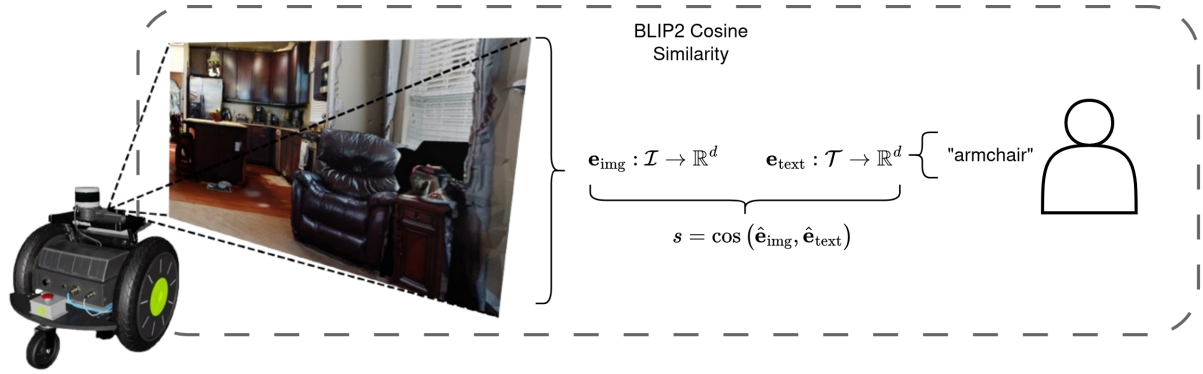

Figure 3: System architecture

Figure 4: Value map example

- Dynamic update of value maps as new observations are integrated.

3.3 Persistent Semantic 3D Mapping

Global Map Construction with Open-Fusion

- Incremental creation of a global semantic point cloud map integrating RGB-D observations over time.
- Registration of observations using robot poses to maintain a consistent world representation.
- Association of semantic labels with 3D points based on query relevance scores.

Semantic Clustering and Graph Node Generation

- Clustering of points with similar semantic labels to form object-level hypotheses.
- Construction of semantic graph nodes representing detected object instances with aggregated confidence scores.
- Maintenance of the semantic graph as a persistent memory for multi-object search tasks.

3.4 Promptable Zero-Shot Detection

- In this work YOLO-E [**yolo_e**] is used as the promptable zero-shot detection model.
- YOLO-E has the following advantages:
  - High inference speed suitable for real-time applications.
  - Ability to handle open-vocabulary object detection based on text prompts.
  - Integration of both visual and textual information for robust detection.
  - Pre-trained on large-scale datasets, enabling zero-shot generalization to unseen object categories.
- Review of traditional and open-vocabulary object detection methods.
- Analysis of grounding-capable detectors and segmentation models for zero-shot tasks.
- Specific evaluation of the following models for their suitability in semantic multi-object search:
  - YOLO-E
  - GroundingDINO
  - MobileSAM
  - GroundedSAM
  - SEEM
  - OWL-ViT
  - MaskDINO
- Discussion of promptable vision-language models supporting multi-modal queries (text, image, audio).
- Challenges with false positives in zero-shot settings and their implications for reliable multi-object detection.

Open-Vocabulary Object Detection with YOLO-E

- Utilization of the YOLO-E model for open-vocabulary object detection based on text prompts.
- Extraction of 2D bounding boxes and associated confidence scores for detected objects.
- Segmentation of detected objects to isolate relevant pixels for 3D localization.

Depth-Based 3D Localization

- With camera intrinsics and depth information, the 2D bounding boxes ans segmentation masks are projected into 3D space.
- Calculation of 3D coordinates for each detected object using depth values within the bounding box.
- Semantic detection pointclouds are passed are then clustered and the centroid of each cluster is computed to obtain robust 3D object locations.
- For each cluster, the mean of the confidence scores of the associated 2D detections is calculated to assign a confidence score to the 3D localization.

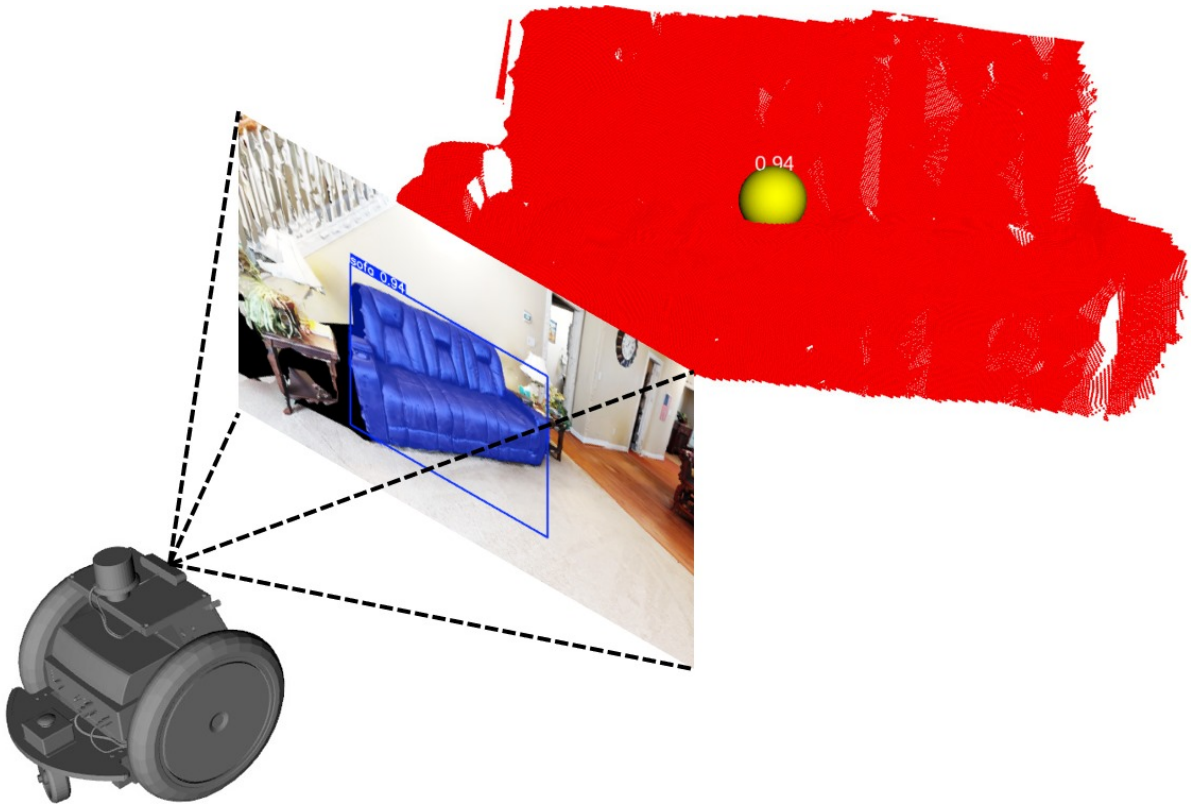

Figure 5: YOLO-E detection to graph node 3D localization

3.5 Fusion Strategy

Exploration–Memory Weighting

- Exploration and memory graph nodes are fused and weighted as follows:
  - Proximity weighting: Nodes closer to the robot's current position are given higher weights, similar to [21].
  - Exploration vs Memory: Nodes from the exploration source are prioritized over memory nodes to encourage discovery of new information, similar to [10].
  - Costmap weighting: Nodes located in areas with lower navigation costs are favored to optimize path planning and navigation efficiency, similar to [21].

Multi-Source Detection Fusion

- Detection graph nodes are weighted based on:
  - YOLO-E confidence scores: Higher confidence detections are given more weight.
  - BLIP-2 value map: Detections with higher semantic relevance to the text prompt are prioritized.
  - The nearer detection graph nodes are to memory graph nodes, the higher their weight.

Relevance Filtering and Node Suppression

- Each source's graph nodes are filtered based on a relevance threshold to eliminate graph nodes within the fov map.
- Relevance map is build over time
- If a graph node is located in an area that has already been explored and found to be irrelevant to the prompt, it is suppressed.

3.6 Behavior Tree for Semantic-Guided Exploration

High-Level Task Structure

- The behavior tree (BT) is designed to manage the high-level task structure for semantic-guided exploration.
- The BT consists of the following main components:
  - Initialization: Clearing Maps, Publishing Prompts

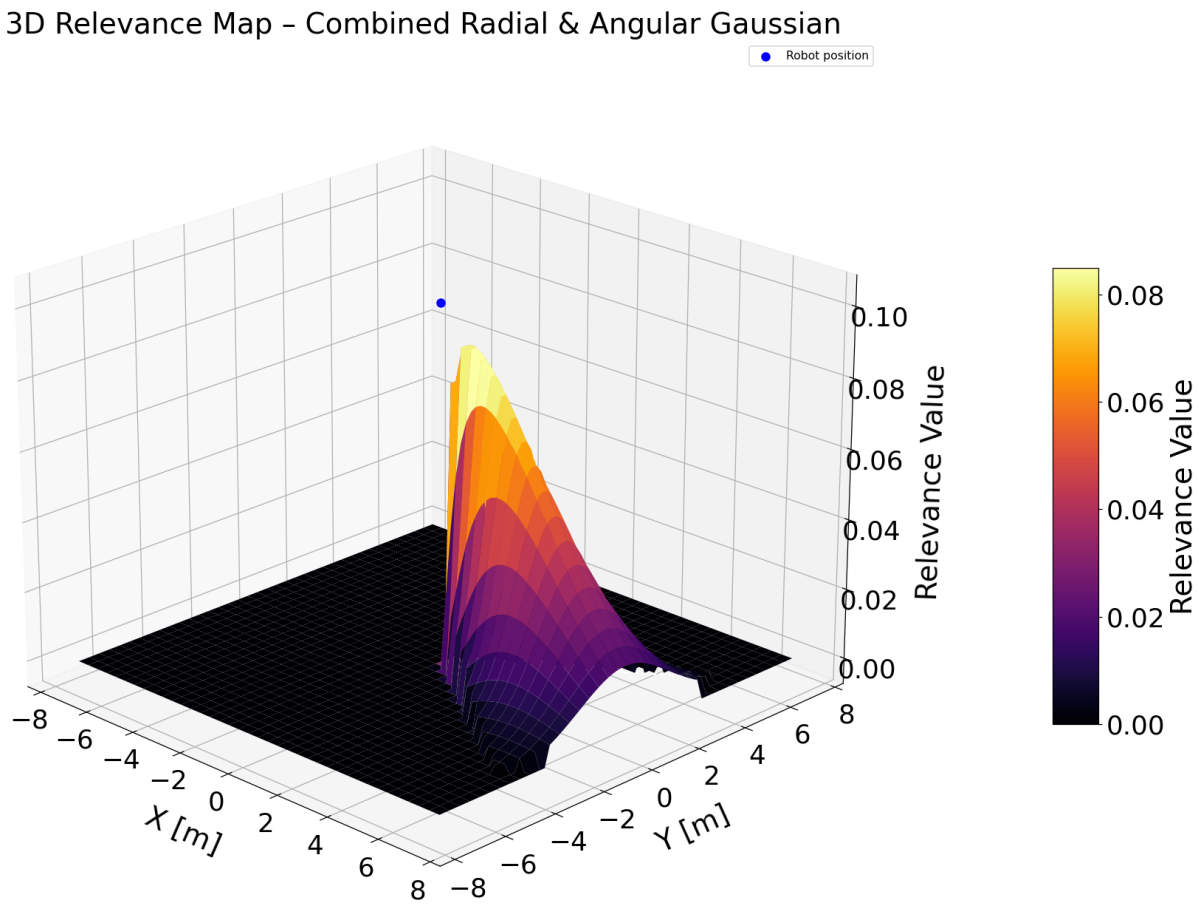

Figure 6: Fusion strategy for exploration, detection, and memory graph nodes

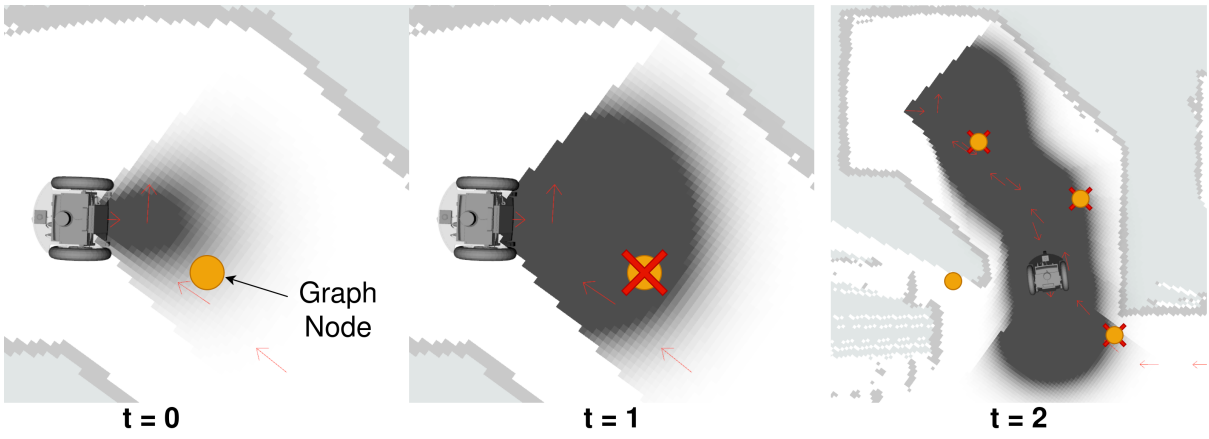

Figure 7: Fusion strategy for exploration, detection, and memory graph nodes

– Detection Branch: If object is detected over a threshold, navigate to it, realign to object take picture
– Exploration Branch: While object not detected, perform semantic frontier exploration navigating to highest valued frontiers or memory nodes
– Termination: If object found, end mission; If time limit reached, end mission
– Behavior tree is called with a ros2 action server, which returns on termination success or failure, and actual path taken

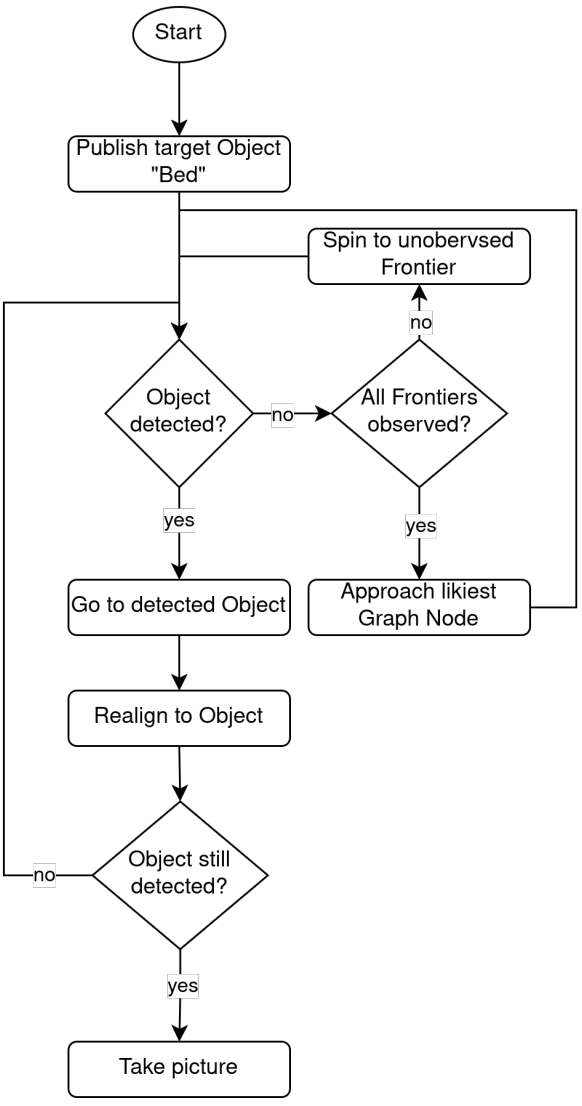

Figure 8: System architecture

Integration with Navigation Stack

- Navigation stack used for low-level path planning and obstacle avoidance.
- Action used: navigate_to_pose, Spin

4 Implementation

This section details the practical implementation of the proposed approach, covering the simulation and real-world setup, datasets, software stack, and hardware configuration.

4.1 Simulation Environment

- Evaluation of simulation frameworks for indoor semantic navigation:
  - HabitatSim: Realistic Matterport3D-based environments with semantic annotations.
  - Isaac Sim / Isaac Lab: GPU-accelerated simulation, advanced physics, support for RTX ray tracing.
  - MuJoCo: High-speed physics engine, limited support for complex indoor scenes.
  - Ignition Gazebo: Modular simulator, ROS2 integration, good for real-robot transfer.
- ...

4.2 Dataset

- Use of **Matterport3D** scenes for realistic indoor environments with ground truth 3D reconstruction and semantic annotations.
- While the Habitat Navigation Challenge 2023 defines Success Rate (SR) and Success weighted by Path Length (SPL) as standard evaluation metrics within the Habitat-Sim environment, this work extends their application to Isaac Sim. Using Isaac Sim allows for a more physically accurate and sensor-consistent setup, incorporating realistic depth noise, lighting variation, and robot dynamics. To ensure comparability, SR and SPL are calculated following the official Habitat definitions, maintaining consistency with prior benchmarks while improving the realism of scene interaction and perception.

4.3 Used Software

- ROS2-based implementation (Humble Hawksbill) as middleware.
- Navigation stack: Navigation2 (Nav2) for frontier-based exploration and path planning.
- DDS communication layer for distributed communication between detection, mapping, and control nodes.

- Integration of promptable models (OpenFusion, BLIP-2, YOLO-E) for real-time zero-shot detection during exploration and exploitation.

4.4 Used Hardware

- **PC:**
  - CPU: AMD Ryzen 9 5950X 16-Core Processor
  - Motherboard: B550 Gaming X V2
  - GPU: ASUS TUF Gaming RTX 4090 24GB OC Edition
  - RAM: 64GB Corsair Vengeance LPX DDR4
- **Real Robot:** Configuration and components to be determined (TurtleBot Waffle).

4.5 Evaluation Metrics

This section defines the evaluation metrics used throughout the experiments and assigns them to each corresponding experiment.

Evaluation Pipeline Overview

- **Semantic Map Generation:** OpenFusion performs semantic segmentation of RGB-D input using the Matterport3D class list. Each segment is assigned its most likely class label from the detection model.
- **Manual Correction:** Incorrectly labeled segments can be manually relabeled within a dedicated ROS 2 node for semantic correction.
- **Data Storage:** OpenFusion saves both the 3D semantic pointcloud and the corresponding 2D SLAM map for each episode. All experiment data follows the `sage_datasets/matterport_isaac` directory structure.
- **Evaluation Initialization:** During evaluation, the saved maps and class definitions are loaded together with a list of target objects (e.g., “bed”, “toilet”, “chair”).
- **Class Filtering and Centroid Extraction:** The evaluator node filters the semantic pointcloud according to the target classes and extracts the 3D centroids of matching clusters.
- **Path Planning:** The shortest-path planner computes the geodesic-optimal path from the robot’s current pose to the nearest centroid of the selected target class, with the Global Path Planner from ROS2 Navigation2.
- **Metric Computation:** The evaluator node compares the planned and executed trajectories to compute Success Rate (SR), Success weighted by Path Length (SPL), and Multi-Object Success Rate (MSR).

- **Result Storage:** Evaluation metrics, trajectories, and intermediate results are stored per episode for analysis and benchmarking.

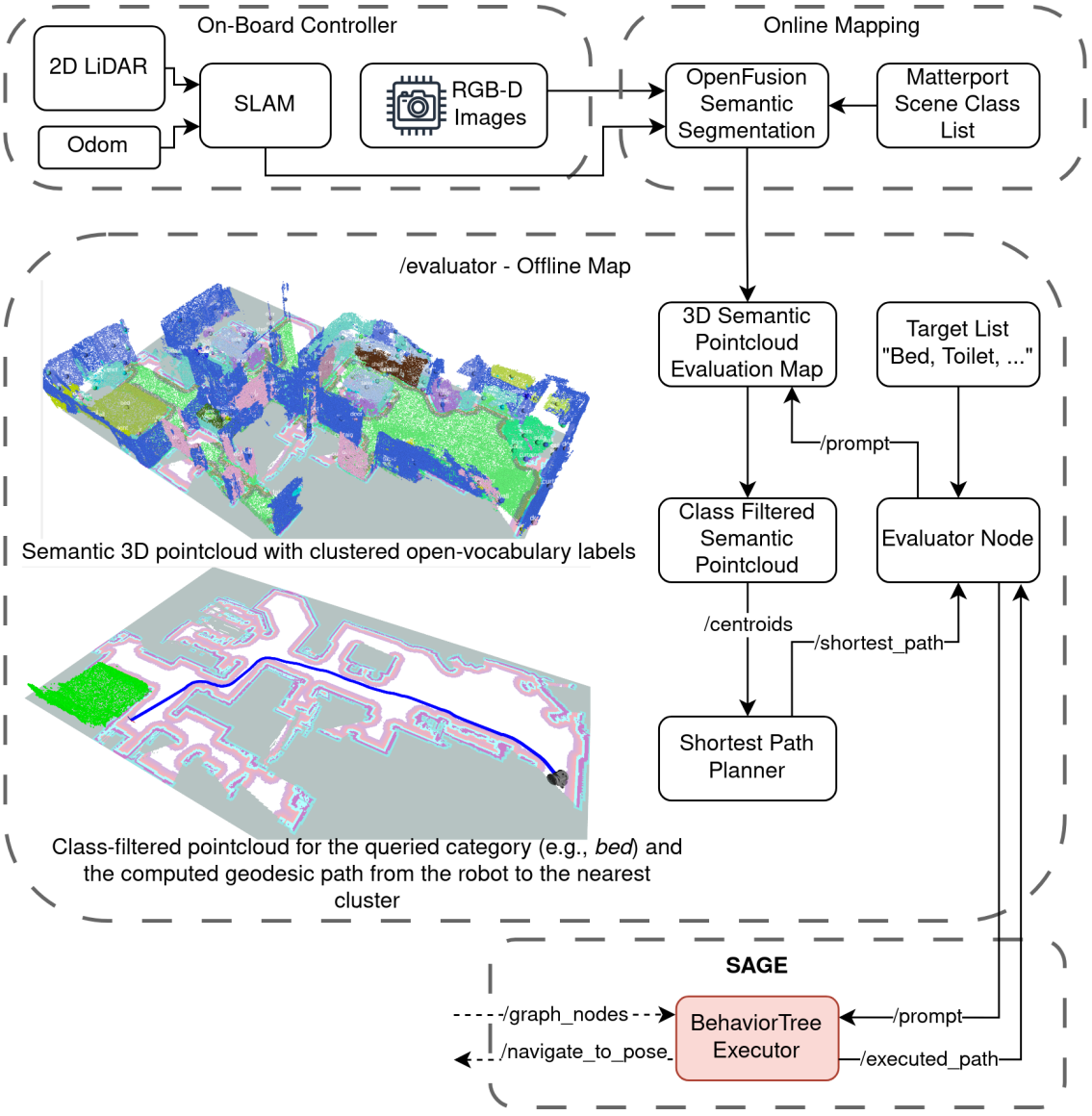

Figure 9: Evaluation pipeline for benchmarking SR, SPL, and MSR in the semantic exploration framework.

Experiment 1: Overall Performance Benchmarking

- Compare SR, SPL and MSR across different baseline methods and the proposed hybrid approach.
- Baselines include:
  - VLFM [**shah2023vlm**],
  - VLMaps [**liu2023vlmaps**],
  - OneMap [**yang20233don**],

  - **Pigeon!** (**Pigeon!**) [**qi2023pigeon**].

- Every scene within the ObjectNav HM3D v2 validation split:
- Within this dataset, each scene contains a set of episodes with the starting pose and target object specified.
- Due to the custom nature of the Isaac Sim environment, all baselines are re-implemented to ensure fair comparison under similar conditions.
- Limitations with IsaacSims Environment setup:
  - Starting Pose Variability
  - Amount of episodes per scene (5 per scene per floor with each 5 sub-episodes for multi-object search)
  - Different requirements:
    * IsaacSim: Photorealistic rendering, physics simulation, realistic sensor noise
    * HabitatSim: Optimized for fast navigation and large-scale datasets
- **Success Rate (SR):** Measures the proportion of tasks in which the robot successfully reaches the queried single goal object. This metric reflects the system's ability to semantically ground a user-specified object and to navigate toward it reliably. It serves as a fundamental indicator of task success and is essential for evaluating overall system effectiveness in basic search scenarios. *Evaluation against:* VLFM, VLMaps, OneMap, GeFF

$$\text{SR} = \frac{1}{N} \sum_{i=1}^{N} S_i$$

  where $S_i = 1$ if the goal was reached in episode i, and 0 otherwise; N is the total number of episodes.

- **Path Efficiency (SPL):** SPL measures the efficiency of successful navigation by comparing the shortest possible path to the actual path taken. It is defined only for successful runs and penalizes overly long trajectories. In the context of semantic exploration, SPL provides insight into how effectively the system prioritizes relevant regions and minimizes detours when searching for target objects.

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

  where S_i is the success indicator for episode i, l_i is the shortest path length to the goal, p_i is the actual path length taken, and N is the total number of episodes.

- **Multi-Object Success Rate (MSR):** The average number of successfully found objects per episode (*Progress, PR*) captures partial success in multi-goal navigation. SPL is computed separately for each object in sequence, conditioned on the success of the

previous one. This highlights the system's ability to reuse semantic map information and improve efficiency across successive targets.

$$\text{PR} = \frac{1}{N} \sum_{i=1}^{N} C_i$$

where C_i is the number of successfully found objects in episode i, and N is the total number of episodes.

Experiment 2: Exploration–Memory Fusion Weighting

- **Objective:** Evaluate how varying the weighting between live semantic exploration and persistent 3D semantic memory influences navigation performance, stability, and overall search efficiency in the hybrid framework.
- **Fusion Parameter:** The trade-off between exploration and memory is controlled through a scalar weighting parameter

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

  which determines the relative influence of frontier-based exploration versus memory-driven exploitation during graph node fusion.
- **Research Questions:**
  - **RQ2a:** How do Success Rate (SR) and Success weighted by Path Length (SPL) vary as the weighting shifts from exploration ($\lambda \to 1$) toward memory-driven behavior ($\lambda \to 0$)?
  - **RQ2b:** Which weighting configuration yields the best trade-off between reactivity (fast adaptation to newly observed information) and stability (robust semantic localization using persistent memory)?
- **Evaluation Procedure:**
  - Multiple runs are conducted across a range of $\lambda_{\text{exploration}}$ values.
  - Performance is measured using the metrics SR and SPL.
- **Expected Outcome:** This experiment highlights whether hybrid fusion provides measurable benefits over purely exploration-driven or purely memory-driven behavior, and identifies the optimal balance for multi-object search tasks.

Experiment 3: Sensitivity to Semantic Map Granularity

- **Objective:** Investigate how the granularity of semantic retrieval in the 3D semantic mapper affects global map quality, navigation performance, and robustness of the hybrid exploration system. Specifically, this experiment evaluates whether dynamic rebalancing between exploration and memory can compensate for increased semantic noise introduced by higher retrieval depths.

- **Semantic Granularity Parameter:** Semantic map quality is controlled through the retrieval depth top-k, which specifies how many semantic candidates (from the VLM embedding space) are fused into each voxel:
  - Low top-k: sharper but potentially incomplete semantics.
  - High top-k: denser semantics but increased noise and ambiguity.
- **Dynamic Fusion Weighting:** To counteract noise introduced by larger top-k values, the exploration weight

  $$\lambda_{\text{exploration}}$$

  is progressively increased, shifting trust toward frontier-driven exploration and away from noisy memory components.
- **Research Questions:**
  - **RQ4a:** How do Success Rate (SR) and Success weighted by Path Length (SPL) degrade as top-k increases while relying primarily on memory?
  - **RQ4b:** Can adaptive rebalancing toward exploration (i.e., increasing $\lambda_{\text{exploration}}$) restore stable performance at higher top-k values?
- **Evaluation Metrics:**
  - SR: ability to consistently reach goal objects under different semantic retrieval granularities.
  - SPL: navigation efficiency and the influence of semantic noise on path quality.
  - Additional qualitative assessment of map sharpness, cluster correctness, and temporal stability of semantic memory.
- **Evaluation Procedure:**
  - Generate semantic maps at multiple top-k levels (e.g., 1, 3, 5, 10).
  - For each top-k:
    - Evaluate SR and SPL under memory-dominant settings.
    - Incrementally increase $\lambda_{\text{exploration}}$ and re-evaluate.
  - Compare results to determine:
    - tolerance of the system to semantic noise,
    - optimal balance between exploration and memory at different granularity levels,
    - interaction effects between map resolution and fusion stability.
- **Purpose:** This experiment analyzes the coupling between semantic map granularity and the stability of exploration–memory fusion. Results reveal how coarse or noisy semantic retrieval affects overall navigation and whether adaptive weighting can maintain robust performance in open-vocabulary mapping environments.

Experiment 4: Robustness to False Positives Through Multi-Source Detection Fusion

- **Objective:** Evaluate how combining multiple semantic evidence sources, instance detection (YOLO-E), semantic similarity scoring (BLIP-2), and memory confidence from the 3D semantic map, improves detection robustness and suppresses false positives during exploration.
- **Fusion Model (Weighted Noisy-OR):** The proposed fusion strategy follows a weighted Noisy-OR formulation, in which independent semantic evidence sources jointly increase the probability of a valid detection:

$$S_{\text{fusion}} = 1 - (1 - w_d S_{\text{det}})(1 - w_c S_{\text{map}})(1 - w_m S_{\text{mem}}).$$

  Here,
  - S_{det}: YOLO-E detector confidence,
  - S_{map}: similarity score from the value map (BLIP-2),
  - S_{mem}: confidence from persistent 3D semantic memory,
  - w_d, w_c, w_m: weights defining the contribution of each source.

  This formulation ensures that high confidence from any source can compensate for uncertainty in others while suppressing spurious detections that lack multi-source agreement.
- **Research Questions:**
  - **RQ3a:** How does overall task performance (SR) change under different weight configurations (w_d, w_c, w_m)?
  - **RQ3b:** How do precision, recall, F1-score, and false-positive rate vary across:
    - COCO-style closed-set categories,
    - open-vocabulary object classes,
    - zero-shot categories not seen during detector training?
  - **RQ3c:** What drawbacks arise when detection thresholds are increased or when a single evidence source is overemphasised (e.g., memory bias, detector hallucination, missed low-confidence but valid detections)?
- **Evaluation Metrics:** Robustness is quantified using classification metrics derived from the confusion matrix:

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

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}},$$

$$\text{FPR} = \frac{FP}{FP + TN}.$$

  Additionally, downstream Success Rate (SR) is recorded for each weight triplet (w_d, w_c, w_m) to evaluate the effect of false positives on the overall navigation pipeline.

- **Evaluation Procedure:**
  - Evaluate a range of weight combinations (w_d, w_c, w_m) spanning detector-dominant, map-dominant, memory-dominant, and balanced regimes, whereas the detection source is mandatory (i.e., $w_d > 0$).
  - Compare against single-source baselines:
    * detector-only (YOLO-E),
    * similarity-only (BLIP-2),
    * memory-only retrieval,
    * the full Noisy-OR fusion strategy.
  - Analyse outcomes under:
    * closed-set (COCO) categories,
    * open-vocabulary targets,
    * zero-shot targets.
  - Quantify how false positives propagate into:
    * erroneous graph node generation,
    * unnecessary navigation actions,
    * degraded SR and SPL.
- **Purpose:** This experiment evaluates whether multi-source, Noisy-OR-based semantic fusion provides a measurable improvement in detection robustness and false-positive suppression compared to single-source methods, thereby enabling more reliable semantic exploration in open-vocabulary indoor environments.

Experiment 5: Real-World System Performance:

- SR, MSR and SP for search performance under real-world conditions.
- System metrics: CPU/GPU usage, FPS, inference latency.

Objective: Assess robustness, efficiency, and deployability in physical environments.

5 Discussion and Results

This chapter presents the experimental evaluation of the proposed hybrid semantic exploration system. Each experiment targets a specific research question and is evaluated using quantitative performance metrics.

5.1 Experiment 1: Benchmarking on Matterport Scenes

Evaluates baseline performance in multi-object search compared to state-of-the-art frameworks (OneMap, VLFM, Pigeon) using SR, SPL, and MSR.

5.2 Experiment 2: Impact of Exploration–Memory Weighting

Analyzes how varying the balance between live exploration and persistent memory influences navigation efficiency and task success.

5.3 Experiment 3: Sensitivity to Semantic Map Granularity

Investigates how varying the semantic retrieval depth affects mapping robustness and overall navigation stability.

5.4 Experiment 4: Effect of Multi-Source Semantic Fusion

Examines how combining detection confidence, semantic similarity, and memory reliability improves detection robustness and reduces false positives.

5.5 Experiment 5: System Efficiency and Real-World Validation

Assesses runtime performance, resource utilization, and stability under real-world sensor noise during physical deployment.

6 Summary and Outlook

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Bibliography

[1] A. Vaswani et al. “Attention Is All You Need.” Comment: 15 pages, 5 figures. arXiv: 1706.03762 [cs]. (Aug. 2, 2023), [Online]. Available: http://arxiv.org/abs/1706.03762 (visited on 12/11/2025), pre-published.

[2] A. Topiwala, P. Inani, and A. Kathpal. “Frontier Based Exploration for Autonomous Robot.” arXiv: 1806.03581 [cs]. (Jun. 10, 2018), [Online]. Available: http://arxiv.org/abs/1806.03581 (visited on 12/11/2025), pre-published.

[3] T.-Y. Lin et al. “Microsoft COCO: Common Objects in Context.” Comment: 1) updated annotation pipeline description and figures; 2) added new section describing datasets splits; 3) updated author list. arXiv: 1405.0312 [cs]. (Feb. 21, 2015), [Online]. Available: http://arxiv.org/abs/1405.0312 (visited on 12/11/2025), pre-published.

[4] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher. “VLFM: Vision-Language Frontier Maps for Zero-Shot Semantic Navigation.” version 1. (2023), [Online]. Available: https://arxiv.org/abs/2312.03275 (visited on 12/11/2025), pre-published.

[5] 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.” Comment: Accepted by/To be published in Robotics: Science and Systems (RSS) 2023; 11 pages, 5 figures. arXiv: 2305.16925 [cs]. (May 26, 2023), [Online]. Available: http://arxiv.org/abs/2305.16925 (visited on 12/11/2025), pre-published.

[6] K. Zhou et al. “ESC: Exploration with Soft Commonsense Constraints for Zero-shot Object Navigation.” arXiv: 2301.13166 [cs]. (Jul. 6, 2023), [Online]. Available: http://arxiv.org/abs/2301.13166 (visited on 12/11/2025), pre-published.

[7] V. S. Dorbala, J. F. Mullen, and D. Manocha, “Can an Embodied Agent Find Your "Cat-shaped Mug"? LLM-Guided Exploration for Zero-Shot Object Navigation,” IEEE Robotics and Automation Letters, vol. 9, no. 5, pp. 4083–4090, May 2024, Comment: 10 pages, ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2023.3346800. arXiv: 2303.03480 [cs]. [Online]. Available: http://arxiv.org/abs/2303.03480 (visited on 12/11/2025).

[8] 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.” arXiv: 2203.10421 [cs]. (Dec. 14, 2022), [Online]. Available: http://arxiv.org/abs/2203.10421 (visited on 12/11/2025), pre-published.

[9] A. Majumdar, G. Aggarwal, B. Devnani, J. Hoffman, and D. Batra. “ZSON: Zero-Shot Object-Goal Navigation using Multimodal Goal Embeddings.” Comment: code: https://github.com/gunagg/zson. arXiv: 2206.12403 [cs]. (Oct. 13, 2023), [Online]. Available: http://arxiv.org/abs/2206.12403 (visited on 12/11/2025), pre-published.

[10] S. K. Ramakrishnan, D. S. Chaplot, Z. Al-Halah, J. Malik, and K. Grauman. “PONI: Potential Functions for ObjectGoal Navigation with Interaction-free Learning.” Comment: 8 pages + supplementary. Accepted in CVPR 2022. arXiv: 2201.10029 [cs]. (Jun. 17, 2022), [Online]. Available: http://arxiv.org/abs/2201.10029 (visited on 12/11/2025), pre-published.

[11] R. Ramrakhya, D. Batra, E. Wijmans, and A. Das. “PIRLNav: Pretraining with Imitation and RL Finetuning for ObjectNav.” Comment: 8 pages + supplement. arXiv: 2301.07302 [cs]. (Mar. 26, 2023), [Online]. Available: http://arxiv.org/abs/2301.07302 (visited on 12/11/2025), pre-published.

[12] 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.” arXiv: 2409.11764 [cs]. (Mar. 3, 2025), [Online]. Available: http://arxiv.org/abs/2409.11764 (visited on 12/11/2025), pre-published.

[13] Q. Gu et al. “ConceptGraphs: Open-Vocabulary 3D Scene Graphs for Perception and Planning.” Comment: Project page: https://concept-graphs.github.io/ Explainer video: https://youtu.be/mRhNkQwRYnc. arXiv: 2309.16650 [cs]. (Sep. 28, 2023), [Online]. Available: http://arxiv.org/abs/2309.16650 (visited on 12/11/2025), pre-published.

[14] D. S. Chaplot, D. Gandhi, A. Gupta, and R. Salakhutdinov. “Object Goal Navigation using Goal-Oriented Semantic Exploration.” Comment: Winner of the CVPR 2020 AI-Habitat Object Goal Navigation Challenge. See the project webpage at https://devendrachaplot.github.io/projects/semantic-exploration.html. arXiv: 2007.00643 [cs]. (Jul. 2, 2020), [Online]. Available: http://arxiv.org/abs/2007.00643 (visited on 12/11/2025), pre-published.

[15] R.-Z. Qiu et al. “Learning Generalizable Feature Fields for Mobile Manipulation.” Comment: Preprint. Project website is at: https://geff-b1.github.io/. arXiv: 2403.07563 [cs]. (Nov. 26, 2024), [Online]. Available: http://arxiv.org/abs/2403.07563 (visited on 12/11/2025), pre-published.

[16] O. Alama et al. “RayFronts: Open-Set Semantic Ray Frontiers for Online Scene Understanding and Exploration.” arXiv: 2504.06994 [cs]. (Apr. 9, 2025), [Online]. Available: http://arxiv.org/abs/2504.06994 (visited on 12/11/2025), pre-published.

[17] C. Huang, O. Mees, A. Zeng, and W. Burgard. “Visual Language Maps for Robot Navigation.” Comment: Accepted at the 2023 IEEE International Conference on Robotics and Automation (ICRA). Project page: https://vlmaps.github.io. arXiv: 2210.05714 [cs]. (Mar. 8, 2023), [Online]. Available: http://arxiv.org/abs/2210.05714 (visited on 12/11/2025), pre-published.

[18] “PIGEON: VLM-Driven Object Navigation via Points of Interest SelectionPreprint. Work in Progress.” (), [Online]. Available: https://arxiv.org/html/2511.13207v1 (visited on 12/11/2025).

[19] I. Lluvia, E. Lazkano, A. Ansuategi, I. Lluvia, E. Lazkano, and A. Ansuategi, "Active Mapping and Robot Exploration: A Survey," Sensors, vol. 21, no. 7, Apr. 2, 2021, ISSN: 1424-8220. DOI: 10.3390/s21072445. [Online]. Available: https://www.mdpi.com/1424-8220/21/7/2445 (visited on 12/12/2025).

[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, Feb. 25, 2021. DOI: 10.3389/frobt.2021.616470.

[21] F. Bourgault, A. Makarenko, S. Williams, B. Grocholsky, and H. Durrant-Whyte, "Information Based Adaptive Robotic Exploration," vol. 1, Feb. 1, 2002, 540–545 vol.1, ISBN: 978-0-7803-7398-3. DOI: 10.1109/IRDS.2002.1041446.

[22] A. Radford et al. "Learning Transferable Visual Models From Natural Language Supervision." arXiv: 2103.00020 [cs]. (Feb. 26, 2021), [Online]. Available: http://arxiv.org/abs/2103.00020 (visited on 12/13/2025), pre-published.

[23] D. Jiang et al. "From CLIP to DINO: Visual Encoders Shout in Multi-modal Large Language Models." arXiv: 2310.08825 [cs]. (Mar. 8, 2024), [Online]. Available: http://arxiv.org/abs/2310.08825 (visited on 12/14/2025), pre-published.

[24] L. H. Li et al. "Grounded Language-Image Pre-training." Comment: CVPR 2022; updated visualizations; fixed hyper-parameters in Appendix C.1. arXiv: 2112.03857 [cs]. (Jun. 17, 2022), [Online]. Available: http://arxiv.org/abs/2112.03857 (visited on 12/15/2025), pre-published.

[25] J. Li, D. Li, S. Savarese, and S. Hoi. "BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models." arXiv: 2301.12597 [cs]. (Jun. 15, 2023), [Online]. Available: http://arxiv.org/abs/2301.12597 (visited on 12/15/2025), pre-published.

[26] S. Liu et al. "Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection." Comment: Code will be available at https://github.com/IDEA-Research/GroundingDINO. arXiv: 2303.05499 [cs]. (Jul. 19, 2024), [Online]. Available: http://arxiv.org/abs/2303.05499 (visited on 12/15/2025), pre-published.

[27] A. Kirillov et al. "Segment Anything." Comment: Project web-page: https://segment-anything.com. arXiv: 2304.02643 [cs]. (Apr. 5, 2023), [Online]. Available: http://arxiv.org/abs/2304.02643 (visited on 12/15/2025), pre-published.

List of Figures

Figure 1 System architecture 15
Figure 2 Frontier detection on occupancy grid 16
Figure 3 System architecture 17
Figure 4 Value map example 17
Figure 5 YOLO-E detection to graph node 3D localization 19
Figure 6 Fusion strategy for exploration, detection, and memory graph nodes 21
Figure 7 Fusion strategy for exploration, detection, and memory graph nodes 21
Figure 8 System architecture 22
Figure 9 Evaluation pipeline for benchmarking SR, SPL, and MSR in the semantic exploration framework. 25

List of Tables

Table 1 Overview of semantic zero-shot and trained exploration approaches. 2
Table 2 Overview of persistent or memory-based semantic mapping approaches. 3
Table 3 Identified gaps in existing semantic exploration frameworks. 4

List of source codes

A Appendix A

B Appendix B