# Where Did I Leave My Glasses? Open-Vocabulary Semantic Exploration in Real-World Semi-Static Environments

Benjamin Bogenberger, *Graduate Student Member, IEEE*, Oliver Harrison, Orrin Dahanaggamaarachchi, Lukas Brunke, *Graduate Student Member, IEEE*, Jingxing Qian, *Graduate Student Member, IEEE*, Siqi Zhou, *Member, IEEE*, and Angela P. Schoellig

***Abstract*—Robots deployed in real-world environments, such as homes, must not only navigate safely but also understand their surroundings and adapt to changes in the environment. To perform tasks efficiently, they must build and maintain a semantic map that accurately reflects the current state of the environment. Existing research on semantic exploration largely focuses on static scenes without persistent object-level instance tracking. In this work, we propose an open-vocabulary, semantic exploration system for semi-static environments. Our system maintains a consistent map by building a probabilistic model of object instance stationarity, systematically tracking semi-static changes, and actively exploring areas that have not been visited for an extended period. In addition to active map maintenance, our approach leverages the map's semantic richness with large language model (LLM)-based reasoning for open-vocabulary object-goal navigation. This enables the robot to search more efficiently by prioritizing contextually relevant areas. We compare our approach against state-of-the-art baselines using publicly available object navigation and mapping datasets, and we further demonstrate real-world transferability in three real-world environments. Our approach outperforms the compared baselines in both success rate and search efficiency for object-navigation tasks and can more reliably handle changes in mapping semi-static environments. In real-world experiments, our system detects 95% of map changes on average, improving efficiency by more than 29% as compared to random and patrol strategies.**

***Index Terms*—Semantic scene understanding, vision-based navigation.**

Received 12 September 2025; accepted 12 January 2026. Date of publication 21 January 2026; date of current version 2 February 2026. This article was recommended for publication by Associate Editor G. Costante and Editor P. Vasseur upon evaluation of the reviewers' comments. This work was supported in part by the German Federal Ministry of Research, Technology and Space (BMFTR) under the Robotics Institute Germany (RIG) with BMBF under Grant 16ME0997K and in part by the EU's Horizon Europe project under the Marie Skłodowska-Curie Actions under Grant 101155035. *(Corresponding author: Benjamin Bogenberger.)*

Benjamin Bogenberger, Oliver Harrison, and Orrin Dahanaggamaarachchi are with the Learning Systems and Robotics Lab, Technical University of Munich, 80333 Munich, Germany, and also with the Munich Institute of Robotics and Machine Intelligence, Technical University of Munich, 80333 Munich, Germany (e-mail: benjamin.bogenberger@tum.de).

Lukas Brunke, Jingxing Qian, and Angela P. Schoellig are with the Learning Systems and Robotics Lab, Technical University of Munich, 80333 Munich, Germany, also with the Munich Institute of Robotics and Machine Intelligence, Technical University of Munich, 80333 Munich, Germany, also with the University of Toronto Institute for Aerospace Studies, North York, ON M3H 5T6, Canada, also with the University of Toronto Robotics Institute, Toronto, ON M5S 0C9, Canada, and also with the Vector Institute for Artificial Intelligence, Toronto, ON M5G 0C6, Canada.

Siqi Zhou is with the Learning Systems and Robotics Lab, Technical University of Munich, 80333 Munich, Germany, also with the Munich Institute of Robotics and Machine Intelligence, Technical University of Munich, 80333 Munich, Germany, and also with Simon Fraser University, Burnaby, BC V5A 1S6, Canada.

Digital Object Identifier 10.1109/LRA.2026.3656790

## I. Introduction

HUMANS can easily navigate unfamiliar environments using prior knowledge and adapt to changes, such as moved furniture or new objects. For example, they might expect to find their reading glasses on a bedside table or near books. Similarly, autonomous robots in everyday environments require semantic understanding, contextual reasoning, and adaptability to changing surroundings.

Object-goal navigation, where a robot must locate objects in unknown or partially known spaces, demands a tight integration of semantic understanding and spatial reasoning. To address this challenge, both modular approaches combining localization, mapping, planning, and control with heuristic exploration strategies [1], [2] and learning-oriented methods [3], [4] have been proposed in the past. More recently, hybrid methods have emerged to improve the exploration efficiency by more seamlessly embedding semantic understanding and reasoning into the pipeline [5], [6], [7]. Yet most works still target static scenes, with any dynamic changes either masked as outliers or tracked only over consecutive frames. Many real-world changes, however, are semi-static (e.g., furniture or electronic devices can be shifted around) and are not directly observed [8], [9]. In such scenarios, maintaining a consistent spatio-temporal environment representation is essential for long-term autonomy and the efficient execution of downstream tasks; this capability, however, remains underexplored in object-goal navigation tasks.

In this work, we propose a semantic exploration framework for semi-static environments (i.e., an environment where objects can be moved, removed, and/or (re)introduced), as illustrated in Fig. 1. Our approach tightly couples mapping in semantic scenes with safe planning and control to leverage the efficiency of modular pipelines, while incorporating language-conditioned semantic understanding and reasoning to enable open-vocabulary, context-aware exploration. Our framework encompasses two modes of operation to support real-world applications: *(i)* active map maintenance, in which the robot actively maintains an up-to-date metric-semantic map by revisiting regions of the map that are likely outdated while deprioritizing areas that are likely static, and *(ii)* open-vocabulary semantic exploration,

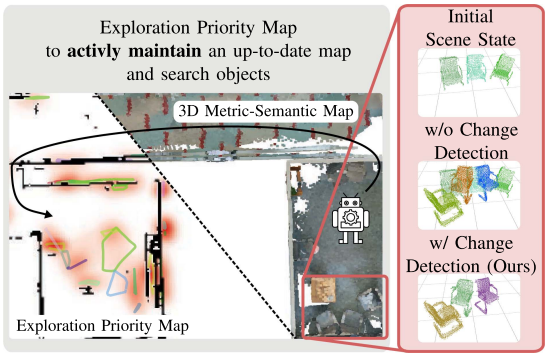



Fig. 1. An illustration of our proposed open-vocabulary semantic exploration approach for semi-static environments, where objects can be shifted, removed, or reintroduced. To account for such changes, the system explicitly maintains a stationarity score for each object instance and actively revisits regions of the map that are likely outdated. This enables the construction of an up-to-date metric-semantic map, which we use to prioritize contextually relevant areas during (unseen) object-goal navigation in semi-static scenes. An overview of our work, including real-world experiments, can be found on our website https://utiasdsl.github.io/semi-static-semantic-exploration/ and in our video http://tiny.cc/sem-explor-semi-static.

in which the robot performs object-goal navigation based on language inputs in unstructured, semi-static environments. Our contributions are summarized as follows:

- We propose a novel online open-vocabulary 3D mapping scheme that incorporates probabilistic change detection to track object instances under possible changes such as being moved, removed, or (re)introduced.
- We introduce a semantic exploration method that allows a robot to actively revisit likely outdated map regions and efficiently perform object-goal navigation in semi-static environments.
- We show that our method outperforms state-of-the-art object-goal navigation and mapping baselines in changing environments through extensive evaluations on public datasets and further validate its efficacy in real-time, closed-loop experiments conducted across three real-world environments.

## II. Related Work

### A. Change Detection in Object-Aware Mapping

While traditional mapping methods have focused on achieving high geometric precision, recent approaches have begun to incorporate semantic understanding to build object-level maps [10], [11], thereby enabling more advanced contextual reasoning and decision-making [12]. A common assumption in localization and mapping systems, however, is that the environment is static, which limits real-world applicability.

Several efforts have emerged to relax this assumption. One common strategy is to identify moving objects and mask them as outliers [13], [14]. While this can reduce artifacts in the environment map, it also leads to information loss and instability when large portions of the scene are dynamic [15]. Another approach is to jointly track the camera pose and dynamic components [16], but such methods often require changes to be observed across consecutive frames. This is limiting in real-world settings, where environment changes are often semi-static, meaning the changes are observed only at discrete times (e.g., when a robot revisits a scene).

A few recent works have attempted to address this challenge. Khronos [17], for example, constructs metric-semantic maps that capture both short- and long-term changes through fragment consistency. However, it associates observations with the map solely based on geometric data, which prevents the tracking of semi-static objects at the instance level. Panoptic TSDF [18] enforces object-level consistency, but its voxel-change counting makes it vulnerable to sensor noise and localization errors. In our work, to maintain a spatio-temporally consistent map, we instead incorporate a probabilistic change detection formulation [8], [9] that fuses semantic prior and geometric consistency to robustly track object instances, even when they disappear and later reappear elsewhere. Moreover, beyond passively bookkeeping changes, our framework closes the perception-action loop to actively improve the map over likely non-stationary regions and consequently enables more efficient completion of downstream tasks.

### B. Semantic Exploration

Semantic exploration refers to the task of using semantic cues to efficiently explore an unknown (or changed) environment [19] to support, for instance, mapping. Non-semantic exploration relies on heuristics to select frontiers [1], [20], while recent semantic approaches guide navigation toward areas likely to reveal relevant space, such as doors [2]. Generative models further extend this by predicting unseen areas to direct exploration [5].

Another line of work deals with object-goal navigation, where the robot is tasked with finding an unseen object of a specific category. Semantic priors are often learned via reinforcement learning (RL) [3], [4], but end-to-end methods lack modularity [19]. This is addressed in [6], [7], which proposes learning top-down semantic maps for use in classical control schemes such as model predictive control (MPC) [6]. However, these methods have several limitations. They are constrained by their training data, often rely on prebuilt maps [3], [6], and lack long-term memory of the space explored [4], [7], [21]. Moreover, all these approaches assume static environments. While methods with short-term memory select frontiers at map edges, we can select internal frontiers to account for potential scene changes.

The open-vocabulary paradigm removes the need for a predefined training-time object set [12], which is made possible by fusing the output of 2D foundation models with 3D information. In open-vocabulary object-goal navigation, the exploration frontiers are selected by, for instance, ranking observed objects using their CLIP embedding similarity to the target [22], [23], or by using LLMs directly to rank frontiers by object descriptions [24], [25]. OK-Robot [26] and ConceptGraphs [12] are closely related to this work, but they cannot deal with changing environments. DynaMem [27] allows updates to its map while also supporting open-vocabulary navigation. However, it can not leverage semantic information to explore towards unseen object goals.

In this work, we propose task-specific open-vocabulary exploration that combines (unseen) object-goal navigation with maintaining an up-to-date map in semi-static scenes. We leverage priors from an LLM and a change detection framework to build top-down semantic maps that guide exploration toward areas of interest–those likely outdated or semantically relevant to the object-goal.

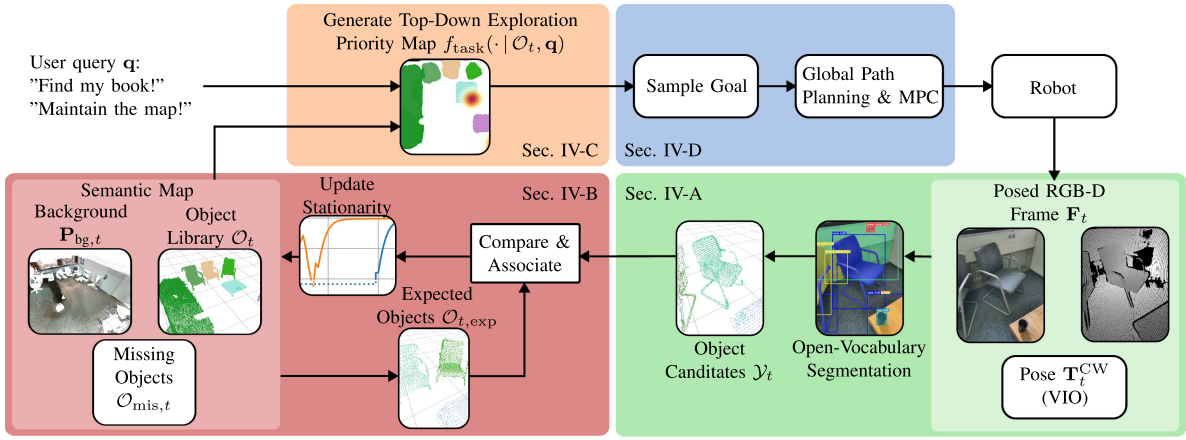



Fig. 2. Overview of our proposed system. We extract object candidates from the current pose and RGB-D frame (green). These are associated with objects in the semantic map, which is updated based on a probabilistic consistency estimate (red). Based on the scene belief, we build a semantic exploration priority map, indicating which map regions are relevant to current tasks – maintaining an up-to-date map or object-goal navigation (orange). Finally, the robot leverages the priority map to select and navigate to sampled positions (blue).

## III. Problem Formulation

Similar to [12], [26], [27], our system takes RGB-D frames $\mathbf{F}_t$ along with camera poses $\mathbf{T}^{\text{CW}}{}_t$, which are assumed known and obtained here via a visual inertia odometry (VIO) system [28]. It incrementally builds and updates a semantic map of a semi-static environment comprising an object library $\mathcal{O}_t$, a missing object library $\mathcal{O}_{\text{mis},t}$, and a background point cloud $\mathbf{P}_{\text{bg},t}$, all of which can be initialized as empty sets or based on a prior map (if available). Given a user query $\mathbf{q}$ (e.g., "Find my glasses!" or "Maintain the map!"), we generate an exploration priority map which allows the robot to either search for the requested object as efficiently as possible or actively revisit likely outdated map regions to maintain a spatial-temporal consistent 3D environment representation.

We note that, similar to other open-vocabulary object-navigation works such as [27], our work focuses on wheeled robots moving on a 2D floor plane. Thus, while we maintain a full 3D map of the environment, a 2D priority map is used for semantic exploration. However, since a 3D map is readily available, our approach can be naturally extended to settings that require full 3D spatial reasoning.

## IV. Methodology

An overview of our method is presented in Fig. 2. At each timestep $t$, we process an RGB-D frame $\mathbf{F}_t$ with its camera pose $\mathbf{T}_t^{\text{CW}}$ to detect a set of currently visible object candidates $\mathcal{Y}_t$; each object candidate $\mathbf{Y}_{t,j} = (\hat{\mathbf{T}}_j^{\text{OW}}, \hat{\mathbf{P}}_j, \hat{\mathbf{f}}_j, \hat{\mathbf{c}}_j)$ comprises, a 6-degree of freedom (DOF) pose $\hat{\mathbf{T}}_j^{\text{OW}}$, a point cloud $\hat{\mathbf{P}}_j$, a visual feature $\hat{\mathbf{f}}_j$, and an open-vocabulary class $\hat{\mathbf{c}}_j$.

The object candidates are used to update the semantic map, comprising the object library $\mathcal{O}_t$, missing object library $\mathcal{O}_{\text{mis},t}$ containing objects that have been explicitly observed to vanish from the scene, and the accumulated background point cloud $\mathbf{P}_{\text{bg},t}$ representing static structures. The object library $\mathcal{O}_t$ and missing object library $\mathcal{O}_{\text{mis},t}$ are two disjoint sets of objects; each object $\mathbf{O}_i = (\mathbf{T}_i^{\text{OW}}, \mathbf{P}_i, \mathbf{B}_i, \mathbf{f}_i, \mathbf{c}_i, \mathbf{s}_i, (t^+{}_i, t^-{}_i), \theta_i)$ comprises, a 6-DOF pose $\mathbf{T}_i^{\text{OW}}$, a point cloud $\mathbf{P}_i$, a visual feature $\mathbf{f}_i$, an open-vocabulary class $\mathbf{c}_i$, a prior stationary label $\mathbf{s}_i \in \{\texttt{static}, \texttt{dynamic}\}$ describing whether objects from class $\mathbf{c}_i$ are typically moved around or not (obtained from an LLM), a first observed time $t^+{}_i$ and latest vanishing time $t^-{}_i$, and distribution parameters $\theta_i$ describing the current belief of the geometric change $l_i$ and stationary score $v_i$, which inform map updates and maintenance. Given a query $\mathbf{q}$ (e.g., "Find my book!" or "Maintain the map!") and scene belief $\mathcal{O}_t$, we build an exploration priority map $f_{\text{task}}(\cdot \mid \mathcal{O}_t, \mathbf{q})$ to sample target waypoints $\mathbf{w}^*$, executed via a global planner and MPC for safe navigation. Each component is further detailed in the respective subsections below.

### A. Current View Object Candidates

Current object candidates $\mathcal{Y}_t = \{\mathbf{Y}_{t,j}\}_{j=1,\dots,J}$ are extracted from the RGB-D frame $\mathbf{F}_t$ and its pose $\mathbf{T}^{\text{CW}}{}_t$ [12]. In particular, the RGB image is segmented using SAM [29] into binary object masks $\mathcal{M}_t = \{\mathbf{M}_{t,j}\}_{j=1,\dots,J}$. Using each mask $\mathbf{M}_{t,j}$ a visual feature vector $\hat{\mathbf{f}}_j$ and class label $\hat{\mathbf{c}}_j$ are computed with CLIP [30], and the RGB-D frame is segmented into point clouds $\hat{\mathbf{P}}_j$, yielding the object candidate $\mathbf{Y}_{t,j}$. Additionally, we filter out observations beyond a maximum distance $d_{\text{max}}$ to the camera. The background point cloud $\mathbf{P}_{\text{bg},t}$ is updated with points not covered by any object mask (i.e., points within $\neg \bigcup_{j=1}^{J} \mathbf{M}_{t,j}$).

### B. Scene Belief Update

The object libraries $\mathcal{O}_{t-1}$ and $\mathcal{O}_{\text{mis},t-1}$ are updated by associating the current object candidates $\mathcal{Y}_t$ with objects expected to be visible in the current camera view and by adding unmatched candidates as new objects. The stationarity scores of all objects are then updated accordingly.

To handle objects that may temporarily disappear or change position across frames (e.g., a chair disappearing and reappearing later elsewhere), object-to-object association is performed within $\mathcal{O}_t$, allowing corresponding $\mathbf{O}_i, \mathbf{O}_j \in \mathcal{O}_t$ to be merged. To avoid exhaustive pairwise comparisons, candidate pairs are selected based on their stationarity scores. These steps are detailed below.

*1) Expected-View Association:* At this stage, we restrict the matching process to objects that are expected to be visible within the current camera's field of view. The goal is to obtain a map

**Algorithm 1:** 2-Step Candidate to Object Association.

**Input:** $\mathcal{Y}_t = \{\mathbf{Y}_{t,j}\}_{j=1,\dots,J}$, $\mathcal{O}_{t,\text{exp}} = \{\mathbf{O}_i\}_{i=1,\dots,I}$
$\mathcal{Y}_{\text{matches}} \leftarrow \varnothing$
**for all** $\mathbf{Y}_j \in \mathcal{Y}_t$ **do** ▷ *Match assuming stationary objects*
  $\hat{\mathbf{O}} \leftarrow \text{argmax}_{\mathbf{O} \in \mathcal{O}_{t,\text{exp}}} S_{\text{geo}}(\mathbf{Y}_j, \mathbf{O})$
  **if** $S_{\text{geo}}(\mathbf{Y}_j, \hat{\mathbf{O}}) > \tau_{\text{geo}} \wedge S_{\text{sem}}(\mathbf{Y}_j, \hat{\mathbf{O}}) > \tau_{\text{sem}}$ **then**
    $\mathcal{Y}_{\text{matches}} \leftarrow \mathcal{Y}_{\text{matches}} \cup \{\mathbf{Y}_j \mapsto \hat{\mathbf{O}}\}$
▷ *Match without assuming stationary objects* ◁
**for all** $\mathbf{Y}_j \in \mathcal{Y}_t \setminus \text{dom}(\mathcal{Y}_{\text{matches}})$ **do**
  $\hat{\mathbf{O}} \leftarrow \text{argmax}_{\mathbf{O} \in \mathcal{O}_{t,\text{exp}} \setminus \text{im}(\mathcal{Y}_{\text{matches}})} S_{\text{sem}}(\mathbf{Y}_j, \mathbf{O})$
  **if** $S_{\text{sem}}(\mathbf{Y}_j, \hat{\mathbf{O}}) > \tau_{\text{sem}}$ **then**
    **if** $\text{RMSE}_{\text{ICP}}(\hat{\mathbf{P}}_j, \mathbf{P}_i) \leq d_{\text{ICP}}$ **then**
      $\mathcal{Y}_{\text{matches}} \leftarrow \mathcal{Y}_{\text{matches}} \cup \{\mathbf{Y}_j \mapsto \hat{\mathbf{O}}\}$
**return** $\mathcal{Y}_{\text{matches}} \cup \{\mathbf{Y} \mapsto \text{None} \mid \mathbf{Y} \in \mathcal{Y}_t \setminus \text{dom}(\mathcal{Y}_{\text{matches}})\}$

from the current object candidates to the expected objects $\mathcal{Y}_t \mapsto \mathcal{O}_{t,\text{exp}} \cup \{\text{None}\}$.

The subset of expected objects $\mathcal{O}_{t,\text{exp}} \subseteq \mathcal{O}_t$ is determined based on the current camera pose $\mathbf{T}_t^{\text{CW}}$ and intrinsics. For each object $\mathbf{O}_i \in \mathcal{O}_{t-1}$, we filter out points in its point cloud $\mathbf{P}_i$ that exceed the maximum distance $d_{\text{max}}$ from the camera, and project the remaining points onto the image plane. This yields the subset $\mathbf{P}_{\text{vis},i} \subseteq \mathbf{P}_i$, representing the object part seen from the current view; the object is considered expected if its visible point fraction exceeds a given threshold, yielding $\mathcal{O}_{t,\text{exp}} = \{\mathbf{O}_i \in \mathcal{O}_{t-1} \big| \frac{|\mathbf{P}_{\text{vis},i}|}{|\mathbf{P}_i|} \geq \tau_{\text{expected}}\}$.

The matching process relies on two similarity measures:

- *Semantic similarity* $S_{\text{sem}} : \mathcal{Y} \times \mathcal{O} \to [0,1]$, defined as the cosine similarity between the candidate's and the object's visual feature vectors $\hat{\mathbf{f}}_j$ and $\mathbf{f}_i$:
$$S_{\text{sem}}(\mathbf{Y}_j, \mathbf{O}_i) = \hat{\mathbf{f}}_j^T \mathbf{f}_i / (\|\hat{\mathbf{f}}_j\|_2 \|\mathbf{f}_i\|_2) \,. \quad (1)$$
- *Geometric similarity* $S_{\text{geo}} : \mathcal{Y} \times \mathcal{O} \to [0,1]$, computed as the fraction of points in the candidate's point cloud $\hat{\mathbf{P}}_j$ whose nearest neighbor in the object's visible point cloud $\mathbf{P}_{i,\text{vis}}$ is within distance $d_{\text{voxel size}}$. With $\hat{\mathbf{P}}_j \cap \mathbf{P}_{i,\text{vis}}$ denoting this subset:
$$S_{\text{geo}}(\mathbf{Y}_j, \mathbf{O}_i) = |\hat{\mathbf{P}}_j \cap \mathbf{P}_{i,\text{vis}}| / \min(|\hat{\mathbf{P}}_j|, |\mathbf{P}_{i,\text{vis}}|) \,. \quad (2)$$

Matching candidates $\mathcal{Y}_t$ to expected objects $\mathcal{O}_{t,\text{exp}}$ (Algorithm 1) first assumes objects are stationary (first `for`-loop), greedily pairing them by geometric similarity, with matches accepted only if both geometric and semantic scores exceed the thresholds $\tau_{\text{geo}}$ and $\tau_{\text{sem}}$, respectively. The second step (second `for`-loop) matches remaining candidates that might have been moved by prioritizing semantic similarity and verifying alignment via iterative closest point (ICP) rather than the geometric measure $S_{\text{geo}}(\cdot,\cdot)$. Matches require semantic similarity above $\tau_{\text{sem}}$ and ICP root mean square error (RMSE) below $d_{\text{ICP}}$. This defers costly ICP until necessary. Finally, matched candidates are merged with their corresponding objects in $\mathcal{O}_t$, and unmatched ones are added as new objects.

*2) Stationarity Score Update:* We maintain a probabilistic stationarity model for every object instance based on [8], which grounds our decision for when to translate or delete objects in our believed scene representation. Each object $\mathbf{O}_i$ contains a belief of its geometric change $l_i \in \mathbb{R}$ and stationarity score $v_i \in [0,1]$ (i.e., $p(v_i, l_i | \dots)$, conditioned as in (3)). Stationarity score $v_i$ represents the likelihood that the object $\mathbf{O}_i$ is still located where last observed, the geometric change $l_i$ measures how much the object has moved since the last timestep (in our case simply the distance its point cloud center of mass has moved).

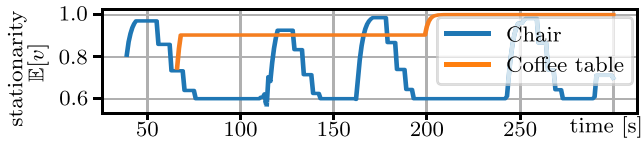



Fig. 3. Different decaying of the stationarity score $\mathbb{E}[v_i]$ for an object with $\mathbf{s}_i = \texttt{dynamic}$ (Chair) and an object with $\mathbf{s}_i = \texttt{static}$ (Coffee table). The objects are observed only at the times their stationary score increases; otherwise, they are not in the robot's view.

At each timestep, we compute the measured change $\Delta_{t,i} \in \mathbb{R}$ of each object (i.e., the distance the object has translated since the last timestep). Based on this change $\Delta_{t,i}$ and the semantic stationarity prior $\mathbf{s}_i$, the belief is updated each step in a Bayesian fashion [8] (we omit the index $i$ here):
$$p(v, l | \Delta_{1:t}, \mathbf{s}) \propto p(\Delta_t | v, l, \Delta_{1:t-1}) p(v, l | \Delta_{1:t-1}, \mathbf{s}) \,. \quad (3)$$
Later, each object's expected stationarity score $\mathbb{E}[v_i]$ informs future updates and map maintenance.

We can directly measure the change $\Delta_{t,i}$ only for objects that have been reobserved. To model the growing uncertainty of out-of-view objects, we inject a synthetic change to $\Delta_{t,i}$ with magnitude defined based on the semantic prior $\mathbf{s}_i$. For a likely `static/dynamic` object, the magnitude is correspondingly smaller/larger; this will correspondingly result in a slower/faster decay in the stationarity score of the object instance. To prevent premature deletion from the scene belief, we stop the decay process once the expected stationarity score $\mathbb{E}[v_i]$ reaches the removal threshold $\theta_{\text{r}}$. Fig. 3 illustrates the evolution of $\mathbb{E}[v_i]$ for a `dynamic` object (chair) and a `static` object (coffee table). The dynamic object is assumed to move more frequently, resulting in a faster decay of its stationarity score.

*3) Object-to-Object Association. Removal, Reintroduction, and Transformation:* In addition to immediate object detection, we must handle objects that disappear from view and may later reappear, potentially at a different location. Removal and translation decisions depend on each object's stationarity $\mathbb{E}[v_i]$, evaluated against the removal and translation thresholds, $0 \leq \theta_{\text{r}} < \theta_{\text{t}} \leq 1$. These thresholds balance system responsiveness and robustness to perception noise.

Objects with $\mathbb{E}[v_i] \leq \theta_{\text{r}}$ are moved from the active object library $\mathcal{O}_t$ to the missing object library $\mathcal{O}_{\text{mis},t}$, where they are kept for potential future reidentification.

We consider two groups of objects for reidentification: objects in $\mathcal{O}_t$ with $\theta_{\text{r}} < \mathbb{E}[v_i] \leq \theta_{\text{t}}$, and objects in the missing library $\mathcal{O}_{\text{mis},t}$. For each object $\mathbf{O}_k$ from these groups that disappeared at time $t^-[k]$, we search for matching objects which appeared in a temporal window $\pm\overline{\tau}$ around its disappearance $\mathcal{O}_{t \approx t^-[k]} = \{\mathbf{O}_i \in \mathcal{O}_t \mid |t^+{}_i - t^-[k]| \leq \overline{\tau}\}$. Matching is performed between $\mathbf{O}_k$ and candidates in $\mathcal{O}_{t \approx t^-{}_k}$ using the same semantically conditioned ICP as in the second `for`-loop of Algorithm 1. If $\mathbf{O}_k$ matches $\mathbf{O}_i \in \mathcal{O}_{t \approx t^-{}_k}$, the two are merged. If the matched counterpart was previously marked missing ($\mathbf{O}_i \in \mathcal{O}_{\text{mis},t}$), it is moved from $\mathcal{O}_{\text{mis},t}$ to the active library $\mathcal{O}_t$ before merging. This enables us to differentiate between objects of the same class and to reintegrate previously observed objects when they reappear (see results in Fig. 9).

### C. Exploration Priority Map

Our navigation framework is based on an exploration priority map, which represents the likelihood of completing the current task $\mathbf{q}$, either object goal navigation or map maintenance, at each 2D point $\mathbf{x} \in \mathbb{R}^2$ in the environment. It is defined as $f_{\text{task}}(\mathbf{x} \mid \mathcal{O}_t, \mathbf{q}) : \mathbb{R}^2 \mapsto [0, \infty)$, with $\int_{\mathbb{R}^2} f_{\text{task}}(\mathbf{x} \mid \mathcal{O}_t, \mathbf{q}) \mathrm{d}\mathbf{x} = 1$, conditioned on the current objects $\mathcal{O}_t$ and on the query $\mathbf{q}$.

We compute the exploration priority map as a superposition of per-object maps $f(\mathbf{x} \mid \mathbf{O}) : \mathbb{R}^2 \to [0, \infty)$, weighted by task-dependent relevancy scores $\lambda(\mathbf{O} \mid \mathbf{q}) \in [0, 1]$, yielding $f_{\text{task}}(\mathbf{x} \mid \mathcal{O}_t, \mathbf{q}) \propto \sum_{\mathbf{O} \in \mathcal{O}_t} \lambda(\mathbf{O} \mid \mathbf{q}) f(\mathbf{x} \mid \mathbf{O})$.

*1) Per-Object Exploration Priority Map:* The per-object exploration priority map $f(\mathbf{x} \mid \mathbf{O}_i)$ is derived by smoothing the object's occupancy shadow with a Gaussian kernel $f(\mathbf{x} \mid \mathbf{O}_i) \propto B_i(\mathbf{x}) * K(\mathbf{x} \mid 0, \sigma(\mathbb{E}[v_i])^2)$ with $\int_{\mathbb{R}^2} f(\mathbf{x} \mid \mathbf{O}_i)\, d\mathbf{x} = 1$, where $B_i(\mathbf{x})$ is a binary mask indicating the ground-plane region occupied (or shadowed) by object $\mathbf{O}_i$. The Gaussian kernel $K(\mathbf{x} \mid 0, \sigma(\mathbb{E}[v_i])^2)$ has zero mean and a standard deviation $\sigma(\mathbb{E}[v_i])$ dependent on the object's stationarity $\mathbb{E}[v_i]$. This smoothing expands the exploration priority map for objects with lower stationarity, reflecting higher uncertainty over their position. We choose the standard deviation $\sigma$ as $\sigma(v) = \frac{v^{-1}-1}{v_{\text{search}}^{-1}-1}(r_{\text{search}} - \sigma_{\text{measure}}) + \sigma_{\text{measure}}$. This models three uncertainty cases: *(i)* as stationarity $v$ approaches 0, uncertainty becomes unbounded (no prior knowledge of location); *(ii)* at an intermediate stationarity $v_{\text{search}}$, a tunable search radius $r_{\text{search}}$ applies; and *(iii)* as $v$ approaches 1, uncertainty is dominated by measurement noise $\sigma_{\text{measure}}$.

*2) Object Semantic Relevancy:* Each object $\mathbf{O}_i$ is scored with a relevancy $\lambda(\mathbf{O}_i \mid \mathbf{q}) \in [0, 1]$ that reflects its importance for the current task. This score is computed differently depending on whether the query $\mathbf{q}$ is the map maintenance task or an object search task.

For map maintenance, the goal is to build an accurate map by prioritizing regions likely to be outdated. Relevancy is based on the object stationarity $\mathbb{E}[v_i]$ that captures both perception confidence and prior stationarity class $\lambda(\mathbf{O}_i \mid \mathbf{q} = \text{maintenance}) = \frac{f_{\text{beta}}(\mathbb{E}[v_i]; \alpha, \beta)}{\max_v f_{\text{beta}}(v; \alpha, \beta)}$, where $f_{\text{beta}}$ is the Beta PDF with parameters $(\alpha, \beta) = (5, 6)$ The Beta distribution is a design choice for mapping stationarity to relevancy: It allows flexible shaping to emphasize low-stationarity objects without overemphasizing highly dynamic ones, but other options are possible.

In object search tasks, the robot locates a target object specified by a language query $\mathbf{q}$. An LLM estimates the likelihood of the target being successful near each known object class $\mathbf{c}_i$ by answering a prompt like "How likely is the query $\mathbf{q}$ successful near a $\mathbf{c}_i$? Return the answer as an integer between 0 and 100, and append a sentence to justify your estimation." separately for each query-class pair $(\mathbf{q}, \mathbf{c}_i)$ pair. The returned value serves directly as the relevancy $\lambda(\mathbf{O}_i \mid \mathbf{q})$, guiding search toward semantically related areas, even for unseen objects. For instance, "Where is my plate?" yields high relevance for tableware, tables, or cabinets (see Fig. 7).

### D. Planning and Control

To this end we want to mimic the priority map $f_{\text{task}}(\mathbf{x} \mid \mathcal{O}_t, \mathbf{q})$ with the robot's infinite-time trajectory $\mathbf{x}_{[0,\infty)}$. While this control problem naturally suggests an ergodic control approach [31], we adopt a sampling-based strategy that enables global path planning, which mitigates the problem of local minima. We navigate to new target waypoints $\mathbf{w}^* \in \mathbb{R}^2$ until the task $\mathbf{q}$ is complete or updated. Navigation is done by planning a collision-free path with $A^*$ and tracking it via MPC.

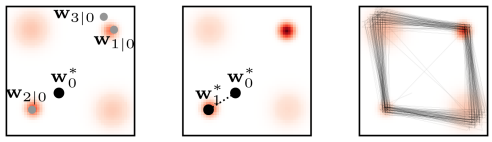



Fig. 4. Illustration of how we sample from the exploration priority map. Last reached target waypoint $\mathbf{w}_0^*$ and iteratively sampled candidate waypoints $\mathbf{w}_{[1,3]|0}$ ($M = 3$) (left). The closest candidate becomes the next target waypoint $\mathbf{w}_1^*$ (middle). Trajectory $\mathbf{w}_{[0,750]}^*$ produced after applying this sampling strategy for 750 steps (right).

To illustrate waypoint selection, assume the $i$-th waypoint $\mathbf{w}_i^*$ was reached at time $t$. Then, the goal is to select the next target waypoint $\mathbf{w}_{i+1}^*$ given the sequence of all past waypoints $\mathbf{w}_{[0,i]}^*$ such that their low-passed density estimate $\hat{f}(\mathbf{x} \mid \mathbf{w}_{[0,i+1]}^*)$ aligns with the exploration priority map $f_{\text{task}}(\mathbf{x} \mid \mathcal{O}_t, \mathbf{q})$. For this, we could sample the next waypoint $\mathbf{w}_{i+1}^*$ directly from the error distribution $e_{i|i}(\mathbf{x}) = f_{\text{task}}(\mathbf{x} \mid \mathcal{O}_t, \mathbf{q}) - \hat{f}(\mathbf{x} \mid \mathbf{w}_{[0,i]}^*)$, which, however, would produce oscillation between modes of the exploration map. Instead, we iteratively sample $M$ candidate waypoints $\mathbf{w}_{[i+1,i+M]|i}$ and select the closest candidate to the robot position $\mathbf{x}_{t|t}$ as the next target waypoint $\mathbf{w}_{i+1}^* = \operatorname{argmin}_{\mathbf{w} \in \mathbf{w}_{[i+1,i+M]|i}} \|\mathbf{w} - \mathbf{x}_{t|t}\|_2$. At each sampling iteration $j = 0, \ldots, M-1$, a new candidate $\mathbf{w}_{i+j+1|i}$ is drawn from the current error distribution $e_{i+j|i}(\mathbf{x})$. This distribution is then updated to $e_{i+j+1|i}(\mathbf{x})$ to incorporate the sampled waypoint, from which the next candidate is subsequently drawn. Fig. 4 illustrates this sampling process.

## V. Experimental Results

We evaluate our approach *(i)* in simulated closed-loop tests on the InteriorAgent [32] dataset, *(ii)* the Khronos [17] mapping dataset, *(iii)* and on Hello Robot's Stretch 3 across two real-world office environments and further validated its generalization in a real-world, object-rich kitchen setting. The robot is equipped with an Orbbec Femto Bolt RGB-D camera and wirelessly connected to a workstation with an NVIDIA RTX 4090. We first present results on datasets comparing directly to state-of-the-art prior work, followed by the real-world experiments. A video showing our experimental results is available at http://tiny.cc/sem-explor-semi-static.

### A. Evaluation on Public Datasets

*1) Object-Goal Navigation Performance:* To evaluate our method against DynaMem [27], we design a set of 60 object-goal navigation tasks in simulation, using eight synthetic interior scenes from the InteriorAgent dataset [32]. These tasks target semi-static environments, for which no standard closed-loop benchmark exists. Each task is categorized based on how the object-goal differs from a prior map generated via frontier exploration: *(i)* the object-goal remains in the same location (Known), *(ii)* the object-goal was absent in the prior map (Novel), and *(iii)* the object-goal has moved (Moved). To introduce changes such as object introduction and movement, we simply hide and reveal object instances of a particular class during the prior map

TABLE I
OBJECT-GOAL NAVIGATION ON THE INTERIORAGENT [32] DATASET

| | Known | | Novel | | Moved | |
|---|---|---|---|---|---|---|
| | SR ↑ | SPL ↑ | SR ↑ | SPL ↑ | SR ↑ | SPL ↑ |
| DynaMem [27] | 0.45 | 0.36 | 0.20 | 0.11 | 0.25 | 0.24 |
| Random | 0.20 | 0.10 | 0.30 | 0.14 | 0.10 | 0.02 |
| Ours | **0.65** | **0.62** | **0.45** | **0.35** | **0.50** | **0.43** |

TABLE II
RESULTS ON THE KHRONOS DATASET. KHRONOS' RESULTS FROM [17]
+: UPPER BOUNDED BY F1 $\leq \frac{1}{2}$(PRE + REC).

| | Method | Dynamics | | | Changes | | |
|---|---|---|---|---|---|---|---|
| | | Pre | Rec | F1 | Pre | Rec | F1 |
| apartment | Khronos [17] | 90.4 | 78.6 | 84.1 | 31.3 | 69.1 | $\leq 50.2^{+}$ |
| | Ours | **92.1** | **86.1** | **88.9** | **94.8** | **84.8** | **89.3** |
| office | Khronos [17] | **96.0** | 59.7 | 73.2 | 24.5 | **54.2** | $\leq 39.4^{+}$ |
| | Ours | 93.8 | **71.3** | **80.2** | **66.5** | 47.0 | **53.0** |

generation phase and the task execution phase. For each task, methods are allocated 15 min to generate a prior map; the scene is then updated, and methods have 5 min to locate the target object. A trial is successful only if the robot reports that it has found the object and is within 1.5 m of it. Table I reports success rate and success weighted by path length (SPL) [33]. Our method consistently outperforms DynaMem [27] and the random-navigation baselines with respect to both metrics. Two representative scenes are depicted in Fig. 5.

*2) Mapping in Changing Scenes:* We evaluate our mapping and change-detection performance on the Khronos dataset [17]. The dataset comprises two synthetic indoor environments–an apartment and a large multi-room office scene–providing RGB-D data, camera poses, ground-truth object annotations, object additions and removals, and human dynamics. Following Khronos, we report the same 4D extensions of precision, recall, and F1-score.

Note that, in Table II, we upper-bound the inconsistent F1 scores reported by Khronos [17] with F1 $\leq \frac{1}{2}$(Pre + Rec), which also holds for the F1 extension to 4D. Our method detects changes with an on average 26.35% higher F1 score, primarily because it incorporates semantic checks, whereas Khronos [17] relies only on geometry. This difference becomes especially clear in the more cluttered apartment scene, where some objects provide only minimal geometric evidence for change. For instance, objects, such as a vase, are too small in scale for reliable geometric change detection, making semantic cues in the image essential. Although the system is not designed for handling dynamic objects, it still improves upon Khronos' Dynamics F1 score by 5.9% on average.

## B. Real World Evaluation

*1) Qualitative Results:* To assess the semantic capabilities of the exploration priority map, we examine two scenarios: the maintenance task (Fig. 6) and the object-goal navigation task involving a plate (Fig. 7). Fig. 6 shows a scene with several mapped objects likely to be moved (chairs and a ball) located outside the robot's current view. The synthetically injected changes cause the objects' stationarity scores to decay over time. As stationarity decreases, these areas gain relevance for the maintenance task, guiding the robot to revisit them. Thus, the robot continually identifies and revisits outdated regions. In Fig. 7, the robot is located in a scene containing everyday

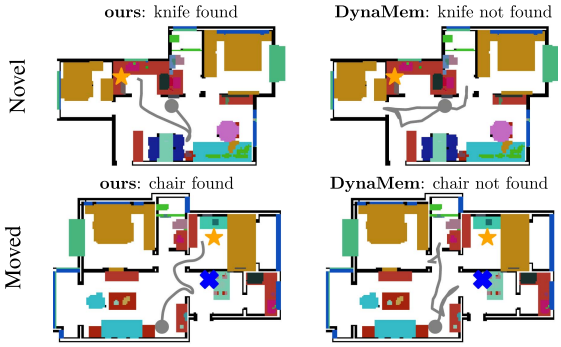



Fig. 5. Example of our method (left) and DynaMem [27] (right) searching for an unseen knife (top) and a moved chair (bottom). Our method checks the dining table, then kitchen and bedroom, while DynaMem explores randomly. Robot path and start shown in gray. Goal object marked with a yellow star; prior location with a blue cross.

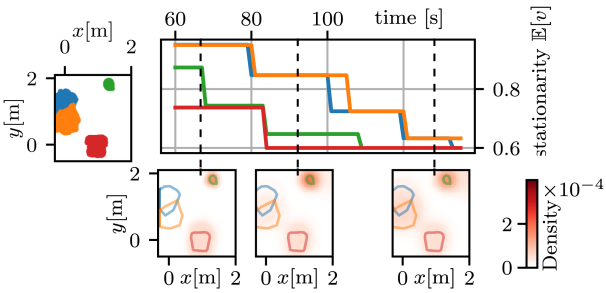



Fig. 6. Exploration priority map evolution over time while maintaining the map. The top left figure shows the map containing four objects. As the objects' stationary score decays (top right), they become more relevant to the map-maintenance task, which is reflected in the exploration priority maps (bottom, from left to right).

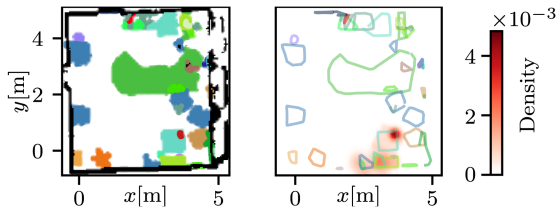



Fig. 7. A plate in an office scene (semantic map on the left) is predicted to be found near a cup ●, coffee table ●, or cabinet ● according to the exploration priority map (right). Other objects in the scene include chairs ● and desks ●.

objects (e.g., chairs, a cup, a coffee table, and a desk) and tasked to find a plate. The exploration priority map correctly infers the plate's likely location near the cup and coffee table. The supplementary video further shows the initial mapping, map maintenance, and object-goal navigation in a real-world `Kitchen`, underscoring the applicability of our method to more object-rich environments.

*2) Semi-Static Mapping Accuracy:* We evaluate mapping accuracy in two environments: a single office and a two-room layout connected by a hallway. Ground truth is obtained via high-precision LiDAR scans and manual object labeling, both before and after scene changes were introduced by moving, adding, and removing objects.

For geometric accuracy, ground truth and mapped point clouds are voxelized at a resolution of 0.1 m, overlaid, and analyzed for true/false positives and negatives. To isolate the impact of changed objects, we limit the comparison to the bounding

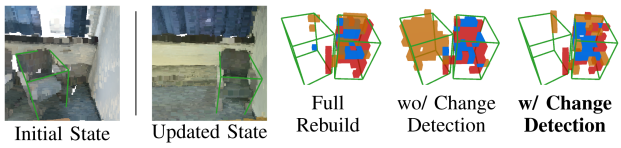



Fig. 8. Voxel-wise comparison to the groundtruth of a mapped chair at initial (left) and updated (right) scene state. The voxels show true positives ■, false positives ■, and false negatives ■. The method without change detection (right second) fails to remove and update the chair, resulting in many false positives.

TABLE III
GEOMETRIC ACCURACY PRE AND POST SCENE CHANGES

| Use Change Detection | Precision ↑ A | Precision ↑ B | Accuracy ↑ A | Accuracy ↑ B | FPR ↓ A | FPR ↓ B |
|---|---|---|---|---|---|---|
| Initial Exploration | 75.9 | 75.3 | 87.1 | 89.0 | 2.0 | 1.9 |
| Full Rebuild | **74.1** | **40.0** | 85.8 | 86.9 | **1.8** | 6.2 |
| ✗ | 44.1 | 30.7 | 83.2 | 83.4 | 9.2 | 12.0 |
| ✓ (Ours) | 71.2 | 38.0 | **87.1** | **87.8** | 2.1 | **5.6** |

A: `Single Office`, B: `Multi-Office + Hallway`

TABLE IV
OBJECT DETECTION ACCURACY PRE AND POST SCENE CHANGES

A: `Single Office` (25 objects), B: `Multi-Office + Hallway` (34 objects)

| Use Change Detection | Precision ↑ A | Precision ↑ B | Recall ↑ A | Recall ↑ B | $F_1$ ↑ A | $F_1$ ↑ B |
|---|---|---|---|---|---|---|
| Initial Expl. | 0.85 | 0.72 | 0.68 | 0.68 | 0.76 | 0.70 |
| ✗ | 0.39 | 0.36 | **0.84** | **0.68** | 0.53 | 0.47 |
| ✓ (Ours) | **0.90** | **0.82** | 0.72 | **0.68** | **0.80** | **0.74** |

boxes of moved objects (see Fig. 8). We compare against three baselines: frontier exploration of the initial scene, full remapping of the changed scene, and a no-change-detection method that cannot update mapped objects. As shown in Table III, the precision drops slightly from initial mapping to updated maps, with a larger drop in the `Multi-Office + Hallway` scenario. Our method performs on par with full remapping and consistently outperforms the no-change-detection method, which exhibits low precision and high false positive rates from failing to remove moved or deleted objects, as depicted in Fig. 8.

Object detection accuracy is measured by comparing mapped object instances using precision, recall, and F1 scores (Table IV). Note that we omit full map rebuilding from this comparison. These results align with the geometric results: no-change-detection yields low precision, whereas our method strikes a balance between precision and recall, as shown by the F1 score. Notably, our method slightly improves over the initial mapping due to more observation data being available.

*3) Semantic Exploration Efficiency:* We evaluate our exploration approach against two baselines: a random policy that selects reachable waypoints at random, and a patrol policy that follows a 2 m grid pattern.

For map maintenance, we measure the ratio of detected to applied changes within a fixed time, counting additions and removals separately (moved objects count as both). Table V shows that in the single-room `Single Office` scenario, all methods perform similarly, but in the larger `Multi-Office + Hallway` scenario, our method outperforms both by quickly passing through the hallway to reach areas with more changes.

Finally, we evaluate our method's effectiveness in locating previously unseen objects. Starting from an initially mapped scene, we place a new object for each run and task the robot with finding it within a maximum allowed time of 2 min 30 s.

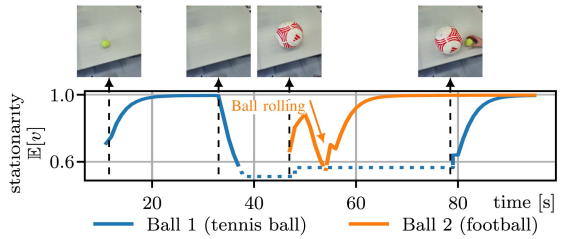
TABLE V
CHANGE DETECTION EFFICIENCY

A: `Single Office` (5 min), B: `Multi-Office + Hallway` (10 min)

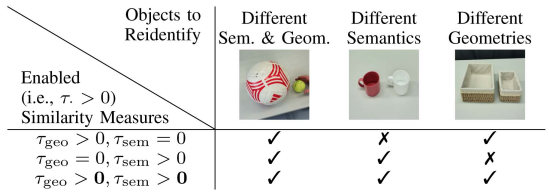
| | Identified Additions (%) A | Identified Additions (%) B | Identified Removals (%) A | Identified Removals (%) B |
|---|---|---|---|---|
| Patrol | 90.9 | 25.0 | 81.8 | 0.0 |
| Random | **100.0** | 37.5 | 90.9 | 40.0 |
| Ours | 90.9 | **87.5** | **100.0** | **100.0** |

TABLE VI
OBJECT GOAL NAVIGATION PERFORMANCE

| | Success Rate ↑ (%) | Mean Success ↓ Time (s) | Weighted Success ↓ Time (s) |
|---|---|---|---|
| Patrol | **100** | 73.75 | 73.75 |
| Random | 25 | **41.50** | 166.00 |
| Ours | **100** | 62.88 | **62.88** |




(a) The stationarity rises and drops when objects appear and disappear.




(b) The table shows which similarity measures are necessary to successfully reidentify the shown object instances. Only when both measures are used all three cases are covered.

Fig. 9. Example of removal, reintroduction, and translation. Our system can distinguish between objects of the same class and reidentify previously shown object instances.

Success requires the robot to stop near and face the target object. Each method is tested in eight runs: 2× books on a shelf, 2× books on a chair, 2× a bowl on a coffee table, and 2× a keyboard on a desk. Table VI reports the success rate $r_s$, mean success time $t_s$, and weighted mean success time $\frac{t_s}{r_s}$. While the random strategy can be faster, it has a success rate of only 25%. Both our method and the patrol strategy are consistently reliable, with ours being ∼14% faster while maintaining a similar success rate for object goal navigation.

## C. Ablation and Sensitivity Analysis

We conduct an ablation study to evaluate the contribution of the semantic and geometric similarity measures ((1) and (2)) to our approach's ability to track object instances over time. We place, remove, and reintroduce two object instances of the same class in front of the robot as shown in Fig. 9(a), and verify whether our system can correctly distinguish the two instances from and each other while still correctly identifying the removed and later reintroduced object. For the two balls, which are both geometrically (their radius) and semantically

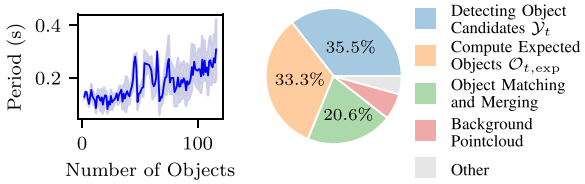



Fig. 10. Map update interval against the number of objects $|\mathcal{O}|$ (left), and relative processing time spent on main mapping components (right).

(their color) different, either similarity measure alone suffices for successful reidentification. For other object pairs (such as the cups and baskets), which differ only in either semantics or geometry, the respective similarity measure must be enabled to distinguish them. Conclusively, the integration of both similarity measures is necessary to robustly track object instances in a semi-static setting.

We analyze our method's sensitivity to parameters on the Khronos [17] dataset. Varying $\tau_{\text{sem}}$, $\tau_{\text{geo}}$, $\theta_{\text{r}}$, and $d_{\text{ICP}}$ over [0.1,0.99], [0.1,0.99], [0.1,0.5], and [0.0001,0.1], respectively, results in a maximum F1 score change of $\pm 7.0$. Overall, the method is fairly robust to parameter changes. We recommend starting with lenient $\tau_{\text{sem}}$, $\tau_{\text{geo}}$, and $d_{\text{ICP}}$ values and increasing them cautiously to avoid false positive detections.

### D. Computation Performance

Fig. 10 (left) indicates that map-update time grows with object count due to per-object operations and the progressively growing background point cloud. However, even with 100 objects, the map updates take only $0.2\,\text{s}$ on average, showing the real-time capability of our approach. A detailed decomposition of the computation is further included in the right panel of Fig. 10.

## VI. Conclusion

In this work, we proposed a novel open-vocabulary semantic exploration approach for robots operating in semi-static environments. Beyond traditional object-goal navigation, our approach actively targets map regions likely to be outdated. We verified its effectiveness against state-of-the art methods, achieving 25% higher SPL and 26% increased change detection F1 in changing scenes. Further, we confirm its transferability to real-world scenarios.

## References

[1] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. IEEE Comp. Intell. Robot. Automat.*, 1997, pp. 146–151.
[2] B. Sun, H. Chen, S. Leutenegger, C. Cadena, M. Pollefeys, and H. Blum, "FrontierNet: Learning visual cues to explore," *IEEE Robot. Automat. Lett.*, vol. 10, no. 7, pp. 6576–6583, Jul. 2025.
[3] D. S. Chaplot, D. P. Gandhi, A. Gupta, and R. R. Salakhutdinov, "Object goal navigation using goal-oriented semantic exploration," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, vol. 33, pp. 4247–4258.
[4] J. Ye, D. Batra, A. Das, and E. Wijmans, "Auxiliary tasks and exploration enable objectgoal navigation," in *Proc. IEEE/CVF Int. Conf. Com. Vis.*, 2021, pp. 16097–16106.
[5] H. Shah et al., "ForesightNav: Learning scene imagination for efficient exploration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2025, pp. 5236–5245.
[6] Y. Goel, N. Vaskevicius, L. Palmieri, N. Chebrolu, K. O. Arras, and C. Stachniss, "Semantically informed MPC for context-aware robot exploration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 11218–11225.
[7] G. Georgakis et al., "Learning to map for active semantic goal navigation," 2022, *arXiv:2106.15648*.
[8] J. Qian et al., "POCD: Probabilistic object-level change detection and volumetric mapping in semi-static scenes," in *Proc. Robot., Sci. Syst.*, 2022.
[9] J. Qian et al., "POV-SLAM: Probabilistic object-aware variational SLAM in semi-static environments," in *Proc. Robot., Sci. Syst.*, 2023.
[10] M. Grinvald et al., "Volumetric instance-aware semantic mapping and 3D object discovery," *IEEE Robot. Automat. Lett.*, vol. 4, no. 3, pp. 3037–3044, Jul. 2019.
[11] N. Hughes, Y. Chang, and L. Carlone, "Hydra: A real-time spatial perception system for 3D scene graph construction and optimization," in *Proc. Robotics: Sci. Syst.*, 2022.
[12] Q. Gu et al., "Conceptgraphs: Open-vocabulary 3D scene graphs for perception and planning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2024, pp. 5021–5028.
[13] C. Yu et al., "DS-SLAM: A semantic visual SLAM towards dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1168–1174.
[14] L. Schmid, O. Andersson, A. Sulser, P. Pfreundschuh, and R. Siegwart, "Dynablox: Real-time detection of diverse dynamic objects in complex environments," *IEEE Robot. Automat. Lett.*, vol. 8, no. 10, pp. 6259–6266, Oct. 2023.
[15] G. D. Tipaldi, D. Meyer-Delius, and W. Burgard, "Lifelong localization in changing environments," *Int. J. Robot. Res.*, vol. 32, no. 14, pp. 1662–1678, Dec. 2013.
[16] B. Xu et al., "MID-Fusion: Octree-based object-level multi-instance dynamic SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 5231–5237.
[17] L. Schmid, M. Abate, Y. Chang, and L. Carlone, "Khronos: A unified approach for spatio-temporal metric-semantic SLAM in dynamic environments," in *Proc. Robot., Sci. Syst.*, 2024.
[18] L. Schmid et al., "Panoptic Multi-TSDFs: A flexible representation for online multi-resolution volumetric mapping and long-term dynamic scene consistency," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 8018–8024.
[19] T. Gervet et al., "Navigating to objects in the real world," *Sci. Robot.*, vol. 8, no. 79, pp. 1–14, 2023.
[20] S. Papatheodorou et al., "Finding things in the unknown: Semantic object-centric exploration with an MAV," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 3339–3345.
[21] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher, "VLFM: Vision-language frontier maps for zero-shot semantic navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2024, pp. 42–48.
[22] J. Jiang, Y. Zhu, Z. Wu, and J. Song, "DualMap: Online open-vocabulary semantic mapping for natural language navigation in dynamic changing scenes," *IEEE Robot. Automat. Lett.*, vol. 10, no. 12, pp. 12612–12619, 2025, doi: 10.1109/LRA.2025.3621942.
[23] S. B. Laina et al., "FindAnything: Open-vocabulary and object-centric mapping for robot exploration in any environment," 2025, *arXiv:2504.08603*.
[24] K. Zhou et al., "ESC: Exploration with soft commonsense constraints for zero-shot object navigation," in *Proc. Intl. Conf. Mach. Learn.*, 2023, pp. 42829–42842.
[25] 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 Robot. Automat. Lett.*, vol. 9, no. 5, pp. 4083–4090, May 2024.
[26] P. Liu et al., "OK-Robot: What really matters in integrating open-knowledge models for robotics," 2024, *arXiv:2401.12202*.
[27] P. Liu et al., "Dynamem: Online dynamic spatio-semantic memory for open world mobile manipulation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2025, pp. 13346–13355.
[28] O. Seiskari et al., "HybVIO: Pushing the limits of real-time visual-inertial odometry," in *Proc. IEEE/CVF Wint. Conf. Applic. Comput. Vis.*, 2022, pp. 287–296.
[29] A. Kirillov et al., "Segment anything," in *Proc. IEEE/CVF Int. Conf. Com. Vis.*, 2023, pp. 4015–4026.
[30] A. Radford et al., "Learning transferable visual models from natural language supervision," in *Proc. Intl. Conf. ML.*, 2021, pp. 8748–8763.
[31] T. Löw, J. Maceiras, and S. Calinon, "drozBot: Using ergodic control to draw portraits," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 11728–11734, Oct. 2022.
[32] M. T. I. SpatialVerse Research Team, "Interioragent: Interactive usd interior scenes for Isaac sim-based simulation," 2025. [Online]. Available: https://huggingface.co/datasets/spatialverse/InteriorAgent
[33] P. Anderson et al., "On evaluation of embodied navigation agents," 2018, *arXiv:1807.06757*.