

Online Structure Learning and Planning for Autonomous Robot Navigation using Active Inference

Daria de Tinguy

`daria.detinguy@ugent.be`

Ghent University

Tim Verbelen

Verses

Emilio Gamba

Flanders Make (Belgium)

Bart Dhoedt

Ghent University

Research Article

Keywords: Autonomous Navigation, Active Inference, Cognitive Mapping, Predictive coding, Topological Navigation, Planning, Robot Navigation, Mobile Robot

Posted Date: July 8th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-9599031/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Online Structure Learning and Planning for Autonomous Robot Navigation using Active Inference

Daria de Tinguy^{1*}, Tim verbelen², Emilio Gamba³, Bart Dhoedt¹

^{1*}Ghent University, Belgium.

²Verses, canada.

³FlandersMake, Belgium.

*Corresponding author(s). E-mail(s): daria.detinguy@ugent.be;

Contributing authors: Tim.verbelen@verses.ai;

emilio.gamba@flandersmake.be; Bart.dhoedt@ugent.be;

Abstract

Autonomous navigation in unfamiliar environments requires robots to simultaneously explore, localise, and plan under uncertainty, without relying on predefined maps or extensive training. We present Active Inference MAPping and Planning (AIMAPP), a framework that unifies mapping, localisation, and decision-making within a single generative model, drawing on cognitive-mapping concepts from animal navigation (topological organisation, discrete spatial representations, and predictive belief updating) as design inspiration. The agent builds and updates a sparse topological map online, learns state transitions dynamically, and plans actions by minimising Expected Free Energy. This allows it to balance goal-directed and exploratory behaviours. We implemented AIMAPP as a ROS-compatible system that is sensor and robot-agnostic and integrates with diverse hardware configurations. It operates in a fully self-supervised manner, is resilient to sensor failure, continues operating under odometric drift, and supports both exploration and goal-directed navigation without any pre-training. We evaluate the system in large-scale real and simulated environments against state-of-the-art planning baselines, demonstrating its adaptability to ambiguous observations, environmental changes, and sensor noise. The model offers a modular, self-supervised solution to scalable navigation in unstructured settings. AIMAPP is available at [link absent for double-blind review]

Keywords: Autonomous Navigation, Active Inference, Cognitive Mapping, Predictive coding, Topological Navigation, Planning, Robot Navigation, Mobile Robot,

1 Introduction

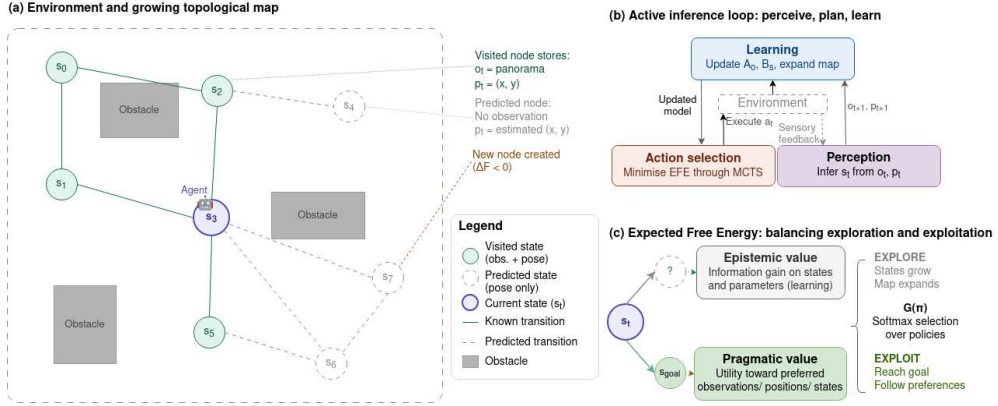


Fig. 1 Overview of the AIMAPP framework. (a) The agent incrementally builds a topological graph; visited nodes (solid) store a panoramic observation and metric position, predicted nodes (dashed) hold only an estimated position. (b–c) Navigation follows a perception–learning–action cycle in which actions are selected by minimising Expected Free Energy, decomposed into epistemic and pragmatic value.

Autonomous mobile robots increasingly operate in environments for which no prior map is available (warehouses being reconfigured, homes the robot has never entered, outdoor spaces with shifting layouts). In such settings, the robot must localise, build a representation of its surroundings, and plan its motion concurrently and online, often under sensor noise, perceptual aliasing and partial observability [1–3]. Each of these subproblems has been studied extensively in isolation, but combining them into a single robust system that does not require pre-training, hand-engineered priors, or globally consistent metric reconstruction remains an open challenge.

Classical approaches address subsets of this problem with characteristic trade-offs. Metric SLAM systems such as ORB-SLAM3 [4] and FAST-LIO2 [5] provide accurate pose estimation but are prone to drift in large or unstructured environments and offer no exploration policy of their own. Path planners such as ETPNav [6] achieve high path quality at substantial computational cost. Frontier-based exploration [7] is lightweight but greedy, often producing redundant backtracking. Learned approaches such as Neural-SLAM [8] improve flexibility through end-to-end training but require extensive offline data collection. Cross-embodiment models such as ViKiNG [9] and NoMaD [10] achieve impressive generalisation to unseen environments but depend on large-scale pre-training or teleoperated demonstrations in the target environment to construct their topological maps before autonomous deployment. These data requirements limit adaptability to genuinely novel settings where no prior experience is available, and no human intervention is desired.

The Active Inference Framework (AIF) [11] offers a different starting point. AIF is a computational theory of perception and action under uncertainty in which an agent

maintains probabilistic beliefs about latent states of the world, predicts the consequences of candidate actions, and selects actions that minimise the expected divergence between predicted and preferred outcomes. AIF is a natural fit for online navigation because it derives both perception and action from the same probabilistic objective: exploration (resolving uncertainty about the world) and goal-reaching (steering observations toward preferences) emerge as two complementary contributions of the same Expected Free Energy (EFE) term, rather than as separate behavioural modes that must be hand-arbitrated.

Building on this idea, we propose AIMAPP (Active Inference MAPping and Planning, Figure 1), a navigation framework that maintains a sparse topological map linking sensory observations to local metric coordinates, and uses it to plan trajectories balancing information gain with goal pursuit. AIMAPP does not rely on pre-training, predefined map structures, or globally consistent metric representations [12]. Instead, it adapts its beliefs continuously from sensorimotor feedback. To keep planning tractable, the agent evaluates only context-relevant policies through Monte Carlo Tree Search (MCTS) over EFE [13]. Localisation is handled probabilistically rather than relying solely on raw odometry, providing resilience to drift in recognisable environments (the mechanism is analysed in Section 3.3, Figure 6). Goals are autonomously assigned and reassessed, and the agent recovers from most motion failures without requiring manual intervention.

AIMAPP exhibits the following key properties:

- **Online learning and self-supervision:** the agent builds and refines its world model from sensorimotor experience, without pre-training or human supervision. This probabilistic formulation provides resilience to sensor drift and tolerates moderate environmental change.
- **Modality and platform flexibility:** any sensor producing comparable observations (RGB, depth, LiDAR) can be integrated, and the model deploys on any ROS-compatible robot.
- **Unified explore/exploit:** exploration and goal-reaching emerge from the same Expected Free Energy objective without task-specific tuning.

We evaluate AIMAPP in simulated and real environments of varying scale and complexity. Across exploration trials, AIMAPP achieves coverage efficiency comparable to or exceeding state-of-the-art planning-based methods (FAEL [14], GBPlanner [15], Frontiers [7]), while achieving coverage efficiency within 74% of human-teleoperated performance on average across all simulated and real-world environments (including Appendix D.2). The system demonstrates robustness to moved obstacles in simulation, sensor drift and sensor uncertainty in the real world, and continues operating in conditions (sandy, carpeted or mixed flooring) where odometry-based baselines such as Frontiers with Nav2 [16] fail outright. Exploration and goal-reaching are achieved within a single architecture, without task-specific re-tuning, across environments ranging from simulated houses and warehouses to real-world garages up to 325 m².

Together, these contributions demonstrate that Active Inference, originally developed in computational neuroscience, can be adapted into a modular and robust

navigation strategy for real robots, in which navigation is treated as hypothesis testing: predictions about future states are generated, evaluated through action, and revised based on surprise.

2 Related Work

Autonomous navigation in unknown environments requires solving several interconnected problems: localisation, mapping, path planning and obstacle avoidance. We review representative approaches below, organised by the trade-offs most relevant to the present work.

Classical navigation methods typically rely on metric mapping via Simultaneous Localisation and Mapping (SLAM). Notable examples include ORB-SLAM3 [4] for visual features, PLP-SLAM [17] for line-and-plane features, and FAST-LIO2 [5] and depth-based SLAM [18] for LiDAR-based pose estimation. These approaches excel in structured, uniformly lit conditions, but are prone to drift over time, even with loop closure, particularly in large or dynamic environments. Furthermore, metric methods often scale poorly in memory and do not integrate planning, as path planning and obstacle avoidance require additional modules [19]. Topological mapping offers a more scalable alternative to metric maps, representing the environment as a graph of connected states rather than dense coordinates [20, 21]. This improves scalability and can better handle perceptual aliasing. Topological representations have been adopted in several learning-based navigation systems: ViKiNG [9] builds a topological graph from offline traversal data for cross-environment transfer, while NoMaD [10] constructs a graph from teleoperated demonstrations in the target environment. In bio-inspired SLAM, RatSLAM [22] builds an experience graph using pose cells and visual templates, achieving robust loop closure without metric consistency, but it provides no autonomous decision-making or goal-directed behaviour. Moreover, most topological methods rely on heuristics or prior data for node creation and are sensitive to environmental change or perceptual noise. AIMAPP inherits the scalability of topological maps while determining node creation through a free-energy criterion rather than a hand-designed heuristic or pre-collected dataset.

Reinforcement learning enables agents to acquire navigation policies through interaction with the environment, improving decision-making under sensor noise and partial observability [2]. Neural-SLAM [8] integrates cognitive mapping with policy learning for visual exploration. Self-supervised methods such as BYOL [23] and RECON [24] learn visual representations without explicit supervision but require months of data collection. World-model methods learn environment dynamics for planning: [3] uses latent Bayesian surprise to drive exploration, while [25] predicts action outcomes to guide policy selection. Active NTFields [26] takes a physics-informed approach, learning arrival-time fields online from depth observations for fast motion planning without pre-training. Bio-inspired neural networks have also been applied to reactive, collision-free navigation [27], though most assume static environments and lack probabilistic reasoning.

These methods are sometimes described as "zero-shot" in the sense that they transfer to new environments without fine-tuning. However, this usage differs from

online structure learning, where an agent discovers the environment’s structure from the first interaction without any prior data collection. ViKiNG requires extensive offline training on diverse environments; NoMaD requires preliminary teleoperation (or fixed policy) in the target environment; our model requires neither. Most learning-based methods are also task-specific, trained for either exploration [8, 23, 24] or goal-reaching [9, 25, 28]. In contrast, AIMAPP requires no pre-training or teleoperated demonstrations, supports any input modality (visual, semantic, point cloud) and unifies exploration and goal-reaching within a single online-learning framework.

AIF treats navigation as inference rather than control: the agent acts to minimise EFE, integrating sensing, prediction, localisation and planning within a single generative process [11]. A comprehensive survey of AIF for robotics is provided by Lanillos et al. [29], covering estimation, control, planning and learning. Early AIF navigation work addressed small grid worlds [30] or simple structured environments [30–32]. Sophisticated inference [33] extended the planning horizon through recursive EFE evaluation, while Fountas et al. [34] demonstrated that deep AIF combined with MCTS can scale to more complex tasks, a combination directly relevant to our MCTS-EFE integration.

More recently, Çatal et al. [35] proposed GSLAM, a SLAM solution learning visual state representations with deep neural networks and constructing topological maps through an experience graph, with validation on a real warehouse robot. Their work showed that localisation and mapping can emerge naturally from free-energy minimisation. Other work has combined AIF with imitation learning for dynamic replanning in simple real-world scenarios [36]. However, both require pre-training over possible actions or observations and do not support exploration in open-ended environments.

AIMAPP differs from these prior AIF approaches in three respects: firstly, it operates in a fully online fashion without pre-training or offline data collection. Secondly, it unifies exploration and goal-reaching within a single generative model rather than addressing one task in isolation; and finally, it demonstrates competitive performance against state-of-the-art classical planning methods (FAEL, GBPlanner, Frontiers) in real environments up to 325 m², a scale not previously addressed by AIF navigation systems. Relative to our own prior work, [13] introduced the discrete-state POMDP and fixed policies integration in small 2D MiniGrid environments without realistic observations, while [37] proposed the hierarchical generative model with learned visual representations. The present paper offers the new (s_t, p_t) decomposition coupling topological and metric information, the online map-expansion mechanism governed by free-energy comparison, the MCTS-EFE integration, the ROS-compatible, sensor-agnostic system architecture, and the experimental evaluation at scale in seven real and simulated environments against non-learning baselines.

3 Method

AIMAPP operates through a recurring sense-predict-learn-act loop formalised under the Active Inference Framework. At each step, the agent 1) infers its current state from sensory observations (perception), 2) simulates possible action sequences and scores them against its beliefs and goals (prediction and action selection), 3) updates its internal model based on the discrepancy between previously predicted and actual

outcomes, as well as currently predicted outcomes (learning) and 4) executes the selected action. This loop is implemented as a partially observable Markov decision process (POMDP), where candidate policies π are evaluated using Monte Carlo Tree Search (MCTS) and scored according to EFE. The system architecture, shown in Figure 2, separates the AIF-based cognitive layer (localisation, mapping, and action selection) from interchangeable perception and motion planning modules, enabling integration with any ROS-compatible platform.

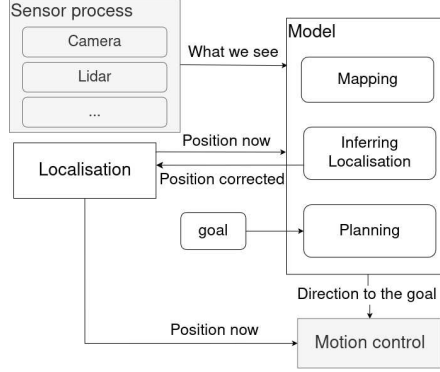


Fig. 2 Overview of the system architecture. Any ROS-compatible solution can replace grey modules. The cognitive layer (localisation, mapping, and action selection) relies on the AIF framework, while perception and motion planning use traditional approaches. Believed odometry takes precedence over sensor odometry. Preferences (goals) are provided by the user when a target observation is to be reached.

3.1 State Representation and Map Structure

The agent maintains a cognitive map, an internal representation inspired by biological navigation systems [12, 38–41]. Cognitive maps support flexible navigation and spatial reasoning, capturing both the layout of the environment and the agent’s experience within it. In our approach, the cognitive map is implemented as a topological graph containing metric information, where nodes represent distinct agent states (s), each corresponding to a spatial location (p) with an associated observation (o). Metric information is retained locally for each node, enabling approximate spatial reasoning without requiring globally consistent coordinates. The map grows adaptively as the agent explores, adding new states when new poses are expected in given directions, supporting scalability to large or dynamic environments.

The latent state is decomposed into s_t (a discrete index representing a node in the topological graph) and p_t (continuous 2D metric coordinates used for local spatial reasoning). The agent observes o_t , in the present implementation a 360° panoramic RGB image used for place recognition and state inference.

The decomposition of the latent variable into topological state s_t and metric position p_t serves both computational and functional purposes. Functionally, this separation loosely mirrors the distinction observed in mammalian navigation systems

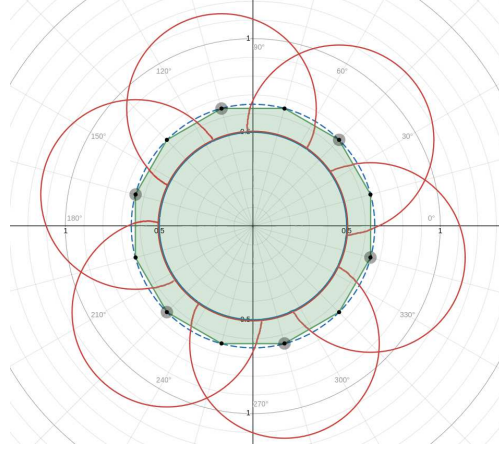


Fig. 3 Influence of a node at position (0,0) on adjacent node creation for an influence radius of 0.5 m and 12 discrete headings spanning 360° . Red circles represent the influence radius of each newly created node. The blue dashed circle marks the robot’s radius, including a padding term to account for its physical size (important near walls). The 12 black dots correspond to the midpoints of the 12 possible action directions. Dots with a black aura indicate valid positions where a new node can be created while respecting the minimum distance constraint (red radius) of adjacent nodes. Dots without an aura represent invalid positions (too close to an existing node) and could instead be created farther away (e.g., at 1 m). This arrangement allows the agent to maintain open junctions for future node expansion.

between place-cell-based location identity and grid-cell-based metric path integration [31, 42]. Computationally, the factorisation enables independent update dynamics: position beliefs evolve continuously through odometry integration, while state beliefs update discretely upon observation and position matching. This separation is particularly advantageous in aliased environments, where identical observations may correspond to distinct locations. By maintaining a separate position estimate p_t , the agent can disambiguate perceptually identical states through accumulated metric displacement, even when observations alone are insufficient [38]. In our setting, a unified representation of p and s into a single distribution would require either discretising metric space (sacrificing localisation precision) or treating place identity continuously (increasing computational complexity without functional benefit) and serve to lead the map expansion over the expected states.

Node creation and connectivity

New nodes are created when a motion leads to a predicted position p_t exceeding a radius of influence from existing states. This expansion step arises from a discrepancy between the current model parameters A_p and expected model parameters $\hat{A}_p$ if the agent were to move to that new position, as defined by the expected information gain on parameters in the EFE (Eq. (6)) and the free energy comparison in Eq. (7).

The discrete set of available actions (e.g., 12 headings plus a “stay” action, the value is user-defined) defines the potential connectivity. This process is illustrated in Figure 3: each state can spawn neighbours only when motion carries the agent outside the influence zone of previous nodes, ensuring sparse but informative coverage.

Observations and place recognition

Observations are stored as 360° panoramas stitched from consecutive camera frames (Figure 4). Each node encodes a coarse spatial position and a panoramic visual observation. Incoming observations are compared to stored visual memories using the Structural Similarity Index (SSIM) to determine familiarity. If no match is found, this signals either a novel location or a significant environmental change.

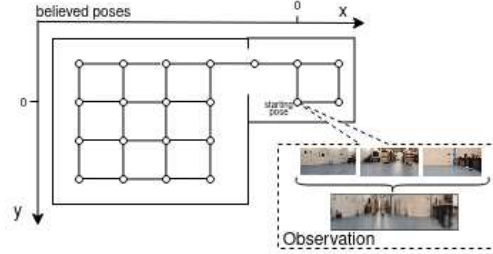


Fig. 4 Schematic of a topological map in a simple two-room environment. Dots are states (nodes), each containing a 360° panorama obtained through stitching images together and associated with a position. Links between nodes are plausible transitions.

Sensory and motion agnosticism

AIMAPP is designed to be modular: it accepts recognised observations from any perception system and relies on a motion controller to report goal-directed stops. This allows interchangeable components within the architecture, as shown in Figure 2. The current implementation uses panoramic images compared via SSIM, but the generative model operates on abstract observation variables and is agnostic to how those observations are derived from raw sensory input. Limitations of the current perception front-end and directions for more robust encodings are discussed in Sections 5.1 and 5.3.

3.2 Generative Model and Inference

World models are internal representations of the environment, generating predictions about possible future sensory information. Classically, worlds are modelled as a POMDP with, at any time t , the current observation o_t and determined past motion a_{t-1} from which the agent can infer state s_t .

Given the decomposition of the latent state into s_t and p_t introduced in Section 3.1, the generative model takes the extended form of Equation (1). Tildes ($\sim$) denote sequences over time.

$$\mathcal{P}(\tilde{o}, \tilde{s}, \tilde{p}, \tilde{a}) = P(o_0|s_0)P(s_0)P(p_0) \prod_{t=1}^{\tau} P(o_t|s_t)P(s_t, p_t|s_{t-1}, p_{t-1}, a_{t-1}) \quad (1)$$

The generative model is parameterised by four distributions, illustrated in the POMDP graph of Figure 5:

- State transitions ($B_s = P(s_t|s_{t-1}, a_{t-1})$): the likelihood of moving between topological states given an action.
- Position transitions ($B_p = P(p_t|p_{t-1}, a_{t-1})$): the likelihood of moving between metric positions given an action.
- Observation likelihoods ($A_o = P(o_t|s_t)$): the probability of observing a particular panorama at a given state.
- Position likelihoods ($A_p = P(p_t|s_t)$): the probability of occupying a particular metric position at a given state.

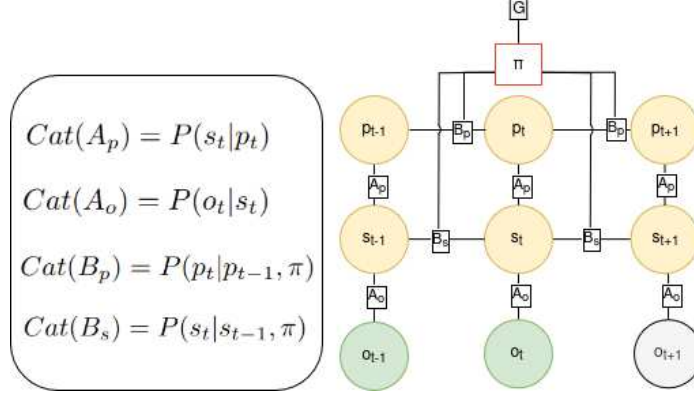


Fig. 5 POMDP of our model. *Cat* stands for Categorical. The model integrates states (s_t), positions (p_t), and observations (o_t) over time, guided by policies (π) and expected free energy (G). The categorical distributions define transition and observation likelihoods: A_p (position likelihoods), A_o (observation likelihoods), B_p (position transitions), and B_s (state transitions). This structure underpins the inference scheme in Equation (2), enabling the agent to infer hidden states and positions from sensory observations and prior beliefs.

In practice, calculating the true posterior $\mathcal{P}$ is typically intractable, because the model evidence $P(o)$ requires marginalising over all latent states [11]. Thus, we resort to variational inference, which introduces the approximate posterior Q . In our temporal model, it takes the form defined in Equation (2). The goal is to make $Q(s)$ as close as possible to $\mathcal{P}(s|o)$, which is achieved by minimising their Kullback-Leibler (KL) divergence.

$$Q(\tilde{s}, \tilde{p}|\tilde{o}, \tilde{a}) = Q(s_0, p_0|o_0) \prod_{t=1}^{\tau} Q(s_t, p_t|s_{t-1}, p_{t-1}, a_{t-1}, o_t) \quad (2)$$

How well this approximation fits the evidence is measured by the Variational Free Energy (VFE) denoted F , defined in Equation (3). Negative F is known as the Evidence Lower Bound (ELBO) in machine learning.

$$F_Q = \underbrace{D_{KL}[Q(\tilde{s}, \tilde{p}|\tilde{a}, \tilde{o})||P(\tilde{s}, \tilde{p}|\tilde{a}, \tilde{o})]}_{\text{posterior approximation}} - \underbrace{\log[P(\tilde{o})]}_{\text{log evidence}} \quad (3)$$

Equation (3) decomposes VFE into two terms: the KL divergence between Q and the true posterior (which measures approximation quality) and the negative log-evidence (which measures how well the generative model explains observations). Although this decomposition references the intractable true posterior, VFE itself can be computed without it: an equivalent expression depends only on the approximate posterior Q and the joint generative model $P(\tilde{o}, \tilde{s}, \tilde{p}, \tilde{a})$, both of which are available by construction. Minimising F therefore simultaneously tightens Q toward the true posterior and increases the model evidence, all without ever evaluating the true posterior directly [11].

3.3 Localisation

Localisation in AIMAPP applies the VFE minimisation introduced in Equation 3 to the topological map. At each step, the agent infers which state s_t best explains the current observation o_t and motion prediction $P(p_t|p_{t-1}, a_{t-1})$ by evaluating the posterior distribution over states, combining predicted motion with sensory evidence. This makes localisation belief-driven: the agent does not depend solely on raw odometry or sensor data, but continuously evaluates which internal state best accounts for its sensory history. When uncertainty over s_t is high, localisation becomes an active process: the agent triggers goal-oriented navigation toward previously observed locations to re-localise.

A key implication of this design is that internal localisation is robust to drift. Since the agent prioritises consistency between predicted and observed outcomes, the exact physical position is less critical than whether its internal model correctly explains the sensory data. In familiar areas, mismatches caused by odometry drift are corrected when the agent revisits a state with a recognisable stored observation; between such events, the belief layer tracks sensor odometry closely. In unexplored areas, however, drift cannot be directly corrected, and the agent must rely on its generative model to maintain consistency until additional information becomes available. If severe drift accumulates before the agent revisits a known region, the resulting map may contain spatially inconsistent nodes, leading to incorrect re-localisation or redundant state creation upon return.

If the agent is confident in its current state s_t (considering VFE against a threshold defined in Appendix A.1), the likelihood matrix A_o is updated to associate the current observation with the believed state, improving robustness against perceptual aliasing and minor visual variations and reducing future surprise. However, if VFE confidence is low, the agent prioritises re-localisation, searching for familiar observations before updating the model.

To illustrate this, consider the scenario in Figure 6. The agent starts at state s_0 with observation o_0 at position p_0 . It intends to move forward 1 m, but due to drift, it actually moves 2 m. Its model, however, creates a new state s_1 at $p_1 = p_0 + 1$ m, consistent with the intended motion rather than the true displacement. After moving

back 1 m (without drift), the agent expects to be at s_0 . At this point, it faces an inference problem: which state best explains the current observation?

Depending on its confidence in motion and observation, four outcomes are possible:

- Figure 6 1) High confidence in motion and observation: the agent recognises o_0 even though it is not exactly at the correct position, infers it is at s_0 , and updates its belief accordingly. Drift remains uncorrected physically, but the model is internally consistent.
- Figure 6 2) Low confidence in motion, high confidence in observation o_1 : the agent recognises o_1 , infers it is at s_1 , and updates its belief over motion to match perception. In this simple example, the drift would be corrected.
- Figure 6 3) Low confidence in both: the agent has high uncertainty about its location. When $P(s_t)$ falls below a user-defined threshold (Appendix A.1), it enters an exploratory phase, seeking consecutive familiar observations to re-localise.
- Figure 6 4) High confidence in position, low confidence in observation: the agent assumes it is at s_0 but fails to recognise the input. It therefore adds a new observation (o'_0) to s_0 , refining the observation model to account for variability (e.g., changes in lighting). The previous observation is not replaced; both are linked to the same state.

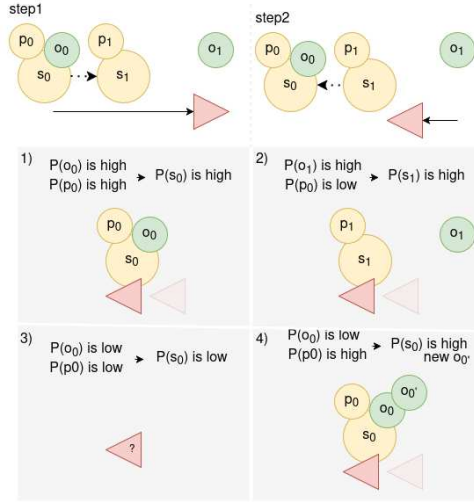


Fig. 6 Impact of drift on the agent’s localisation. The top row illustrates the scenario: the red triangle is the agent; the solid line shows the true trajectory; the dashed line shows the trajectory perceived through odometry; the yellow circles are inferred states; the green circles are observations. The agent intends to move 1 m right from s_0 but actually moves 2 m. Its model creates state s_1 at the perceived position, while the true observation o_1 belongs to the real position. On the next step, the agent moves 1 m left (without drift) and expects to return to s_0 . Four localisation outcomes are then possible: 1) High confidence in both: recognises o_0 , infers s_0 , drift physically uncorrected. 2) Low motion confidence, high observation confidence: recognises o_1 , infers s_1 . 3) Low confidence in both: enters exploratory phase. 4) High position confidence, low observation confidence: adds o'_0 to s_0 .

3.4 Planning and Decision-Making

According to the Free Energy Principle, agents must minimise free energy to form a model that best explains their environment [11]. For perception, this is implemented by minimising VFE during inference, as described in Section 3.2. For action and planning, the same principle extends into the future: agents should act to minimise the Expected Free Energy G under their candidate policies.

Formally, a policy π is a sequence of future actions. The EFE $G(\pi)$ quantifies how well following π is expected to reduce surprise [30]. It decomposes into two complementary contributions:

- **Exploration (epistemic value)**: expected information gain reducing uncertainty about the environment.
- **Exploitation (pragmatic value)**: expected alignment with preferred observations, such as reaching a specific target.

This decomposition illustrates that exploration and exploitation are not separate objectives bolted onto the model, but emerge naturally from minimising EFE. This parallels a rat in a maze [43], which alternates between venturing into unknown corridors (uncertainty reduction) and returning to rewarding locations (preference fulfilment).

Candidate policies are generated from the current state through MCTS [34, 44]; the algorithm definition and pseudo-code are provided in Appendix A.3. This approach incrementally builds a search tree by simulating possible action sequences and focusing computation on the most promising branches, considering EFE over policies. This yields three advantages over predefined or exhaustive policy sets: (1) scalability, as it avoids combinatorial explosion in large action spaces; (2) adaptability, as it dynamically prioritises policies relevant to the agent’s current context; and (3) efficiency, as it limits computation to policies that meaningfully differ in their predicted outcomes.

To determine the most relevant policy at any time t , all policies are assessed through a softmax transformation (Eq. (4)):

$$P(\pi) = \sigma(-\gamma G(\pi) - H) \quad (4)$$

where σ is the softmax function and γ is a precision parameter (inverse temperature) controlling action stochasticity. Higher values of γ bias the agent toward policies with minimal expected free energy, while lower values promote greater randomness in action selection.

A general issue with prediction over future steps through probability distribution, without relying on hierarchical structures, is that the further into the future the prediction lies, the more diluted the probabilities become. To address this, we introduce an inductive term H [45] in the policy evaluation. This term propagates pragmatic value from preferred states beyond the horizon range of the policies back to the current state, adding weight to actions leading toward those states. For a detailed derivation of this term, refer to Appendix A.2.

While the inductive prior H propagates utility from distant states back to the current decision point, it does not fully resolve the finite horizon limitation inherent

to AIF and MCTS. Limiting search depth means that rollouts do not reach terminal states, and standard guarantees on sample mean convergence do not strictly apply. In practice, this may reduce action selection accuracy for objectives separated by many state transitions from the current position. Hierarchical planning [37] offers a principled extension for such scenarios; we leave this integration to future work.

The total EFE of a particular policy is the sum over its constituent time steps (Eq. (5)):

$$G(\pi) = \sum_{\tau} G(\pi, \tau) \quad (5)$$

where τ indexes consecutive steps of the future policy.

The per-step EFE $G(\pi, \tau)$ balances epistemic values (information gain over future states and parameters) with pragmatic values (preference satisfaction), and additionally accounts for collision avoidance. In our formulation, the collision perception c is decoupled from the observation o for clarity. Although collision information is fundamentally derived from the observation, separating it allows the model to treat obstacle avoidance as an independent pragmatic constraint. In practice, c_τ is obtained by filtering 2D LiDAR range readings into a binary signal per action direction (obstacle present or absent). The model treats this as a generic depth feature; any sensor providing directional obstacle information (e.g., a depth camera or stereo pair) could serve as the source without altering the decision-making process. The probability $P(c_\tau)$ is evaluated against the preference C_c of not encountering an obstacle, which has a major impact on policy viability. The full decomposition is given in Equation (6):

$$\begin{aligned} G(\pi, \tau) &= \mathbf{E}_{Q_\pi} [\log Q(s_\tau, p_\tau, A_p | \pi) - \log Q(s_\tau, p_\tau, A_p | c_\tau, \pi) \\ &\quad - \log P(c_\tau | C_c) - \log P(o_\tau | C_o) \\ &\quad - \log P(p_\tau | C_p) - \log P(s_\tau | C_s)] \\ &= \underbrace{\mathbf{E}_{Q_\pi} [\log Q(s_\tau, p_\tau | c_\tau, o_\tau, \pi) - \log Q(s_\tau, p_\tau | \pi)]}_{\text{expected information gain on states (inference)}} \\ &\quad - \underbrace{\mathbf{E}_{Q_\pi} [\log Q(A_p | s_\tau, p_\tau, c_\tau, o_\tau, \pi) - \log Q(A_p | s_\tau, p_\tau, \pi)]}_{\text{expected information gain on parameters (learning)}} \\ &\quad - \underbrace{\mathbf{E}_{Q_\pi} [\log P(c_\tau | C_c)]}_{\text{expected collision}} - \underbrace{\mathbf{E}_{Q_\pi} [\log (P(o_\tau | C_o))]}_{\text{utility term on observation}} \\ &\quad - \underbrace{\mathbf{E}_{Q_\pi} [\log (P(p_\tau | C_p))]}_{\text{utility term on position}} - \underbrace{\mathbf{E}_{Q_\pi} [\log (P(s_\tau | C_s))]}_{\text{utility term on state}} \end{aligned} \quad (6)$$

The per-step EFE $G(\pi, \tau)$ decomposes into six terms summarised in Table 1 and detailed in Equation (6).

In Equation (6), the expectation $\mathbb{E}_{Q_\pi}[\cdot]$ is taken over the predicted distribution of future states, positions, observations, and collisions under policy π . The position likelihood A_p associates each state with its metric position; rather than maintaining a parametric distribution over A_p , the model stores positions directly and evaluates spatial consistency when predicting state transitions (see Section 3.1 and Appendix A.1).

Table 1 Short description of the per-step EFE terms.

Term	Type	Role
Info. gain on states	Epistemic	Reduces uncertainty over s_τ, p_τ
Info. gain on parameters	Epistemic	Improves the position model A_p
Expected collision	Pragmatic	Penalises paths toward obstacles
Observation preference	Pragmatic	Attracts toward target observations
Position preference	Pragmatic	Attracts toward target positions
State preference	Pragmatic	Attracts toward target states

The expected information gain quantifies the anticipated shift in the agent’s belief over the state from the prior (e.g., $Q(s_\tau|\pi)$) to the posterior (e.g., $Q(s_\tau|c_\tau, o_\tau, \pi)$) when pursuing a particular policy π . The utility terms assess the expected log probability of observing preferred outcomes under the chosen policy. This value measures the likelihood of the policy guiding the agent toward its preferences (e.g., avoiding collisions, reaching target observations, positions, or states). Rather than representing “reward” in the reinforcement learning sense, these terms measure coherence between predicted outcomes and the agent’s prior preferences, encoded in the distributions C_c , C_o , C_p , and C_s .

3.5 Learning and Map Expansion

Having determined how the agent infers its state and selects actions, we now describe how it updates its internal parameters in light of inferred poses and current and expected observations. This learning step closes the perception-learning-action loop, enabling the agent to refine its world model so that it continues to explain sensory observations accurately.

The agent compares its current generative model, as defined by parameters governing pose likelihoods (A_p), observation likelihoods (A_o), and state transitions (B_s), against an expanded model incorporating newly predicted or observed states [13]. Whether to expand the map is determined by comparing the free energy of the current model against that of an expanded model. This comparison is evaluated over the pose likelihood A_p (Eq. (7) [46, 47]): if the expanded model achieves lower free energy ($\Delta F < 0$), the map is expanded by adding the new state to A_p , and B_p , B_s and A_o are resized accordingly to match the new state dimensionality. The transition model B_s and observation likelihood A_o are additionally updated independently of map expansion: B_s is refined via Dirichlet pseudo-counts whenever transitions are observed, and A_o is updated when localisation confidence is high.

$$\Delta F = F[\hat{A}_p(\theta)] - F[A_p(\theta)] \quad (7)$$

Here, $\hat{A}_p$ represents the updated pose likelihood model, and θ its parameters. If ΔF is negative (meaning the new model has lower free energy), the agent updates its internal structure A_p to incorporate the newly predicted information.

When a new state is predicted, the pose model A_p is expanded, triggering corresponding updates to the pose transition tensor B_p and to all other state-related matrices. The updated observation model A_o assigns uniform probabilities to unvisited

states, reflecting initial uncertainty (high entropy, low $P(s_t)$). The transition model B_s is also adapted to connect new states with existing ones. The Dirichlet pseudo-count update on state transition has learning rates depending on whether the trajectory was imagined or physically experienced, and whether it was deemed feasible or impossible. Concisely, an experienced transition reinforces the transition belief more than a predicted transition; inversely, a physically experienced obstruction reduces the transition belief more strongly than a predicted obstacle (details in Appendix A.1 Table A2).

A similar expansion mechanism applies during inference when a novel sensory input is encountered. In this case, only A_o is expanded, since the update pertains to the observation dimension rather than the state dimension.

By continuously applying this update process, the agent maintains an adaptable, self-consistent world model integrating both real and imagined experience. This ensures that exploration not only extends the map but also improves the accuracy of navigation when pursuing specific goals.

3.6 Worked Example

To illustrate the perception-learning-action cycle concretely, consider the following scenario from exploration. At timestep t , the agent believes it occupies state s_3 with position $p_t = (2.1, 3.4)$ m in its local coordinate frame. The topological map currently contains one visited node and 5 nodes in total. The observation likelihood matrix A_o is a 5×5 distribution associating the first node’s stored panorama with its corresponding state. The rest of the matrix is filled with near-zero values (high uncertainty).

The agent evaluates candidate policies via MCTS (depth 10, 30 simulations) and selects action $a_t = \text{heading}_7$ (moving approximately 210° relative to the robot’s global orientation). After executing this motion, odometry reports a displacement of 0.6 m, yielding a predicted position $p_{t+1} = (1.6, 2.9)$ m.

Upon arrival, the agent captures a new 360° panorama o_{t+1} and computes SSIM scores against all stored observations. Suppose the highest SSIM score is 42%, falling below the recognition threshold (set to 65% in this work). Simultaneously, the predicted position p_{t+1} lies within the 0.5 m influence radius of one unvisited node s_5 . Together, the posterior belief concentrates on the new state: $Q(s_{t+1} = s_5) \approx 1$. Considering the depth of the observation (obtained from LiDAR) and the influence radius of the current node, the model A_p expands in all free directions beyond the existing nodes’ influence radius from the current believed state s_5 . New states s are created to match the new dimensionality of A_p in B_p . The transition matrix B_s and likelihood A_o expand in proportion to match the new map dimension. The transition matrix B_s is updated via Dirichlet pseudo-counts to reflect the newly traversed edge $(s_3, a_t) \rightarrow s_5$ and the believed connections between nodes given by the current observation.

4 Results

We evaluate AIMAPP across simulated and real-world environments, assessing exploration efficiency, robustness to drift and environmental changes, and goal-reaching performance. Computational scalability is reported at the end of this section.

4.1 Experimental Setup

Environments

Experiments were conducted in four simulated and three real-world environments. The simulated setups comprised three warehouses of increasing size: mini (36 m²), small (80 m²), and large (280 m²), based on the Amazon Gazebo environment [48], featuring aisles, boxes, and industrial obstacles such as forklifts. A 175 m² house environment [49] without doors, containing kitchens, playrooms, and bedrooms, was also used. Both warehouses and the house included objects that are challenging for LiDAR-based detection, such as curved chairs and forklift legs. Real-world experiments were performed in three settings: (1) a small, fully controlled 20.3 m² bedroom with drift-inducing flooring (carpet and wooden floor), (2) a controlled 185 m² warehouse equipped with Qualisys motion-capture cameras for ground-truth odometry, and (3) a large 325 m² sandy parking lot where moving cars occasionally altered the navigable space (navigation was paused while cars crossed). Environment layouts are provided in Appendix C.

Baselines

We compare AIMAPP against several exploration strategies that do not require pre-training:

- **Frontier-based exploration** [7], using a 2D LiDAR and Nav2 SLAM [50].
- **GBPlanner**, an enhanced version of the 2021 DARPA SubT Challenge winner [51], combining Voxblox [52] with a topological map for 3D exploration planning. It operates with three cameras and two 3D LiDARs.
- **FAEL** [14], based on frontier logic, using 3D mapping via UFOMap [53] and topological navigation. It relies on a 3D LiDAR.
- **Manual exploration**, in which a human operator teleoperated the robot while Nav2 SLAM cartographed the surroundings. The human has prior knowledge about the environment.

For fairness, all models operated with a maximum sensing range of 12 m regardless of sensor dimensionality. Because AIMAPP is a zero-shot learning agent, in the sense that it does not train over any prior data, we did not compare against learning-based methods [9, 10, 23, 24] requiring pre-training, which would have had prior knowledge of the environments or required a pre-mapping. Details about robots, sensors and each baseline configuration are provided in, respectively, Appendix B and D.1.

Metrics

We report the following metrics throughout this section. *Coverage Efficiency* (CE) is defined as the area covered (in m²) divided by the distance travelled (in m); time is excluded, as it depends on motion-planning parameters (wheel speed) rather than the decision-making process. *Normalised Area Under the Coverage Curve* (nAUC) captures the progression of coverage relative to distance, with higher values indicating faster convergence to full coverage. *Root Mean Square Error* (RMSE) over the (x, y) axes quantifies positional drift with respect to ground-truth odometry. Additionally,

we report the trajectory-length-normalised drift, computed as the absolute trajectory error per metre travelled (ATE/m), to enable comparison across runs of different lengths. For goal-reaching, we report the ratio of the agent’s travelled path length to the optimal path computed by A* over the agent’s topological graph (ideal path given agent knowledge), which assumes perfect knowledge of transitions (which implies that the A* can imagine passing through an impossible transition if it was deemed traversable or, instead, miss a shortcut) and an A* over an occupancy map of the environments (most ideal path). All metrics are averaged over a minimum of five runs per condition, with standard deviations reported.

Protocol

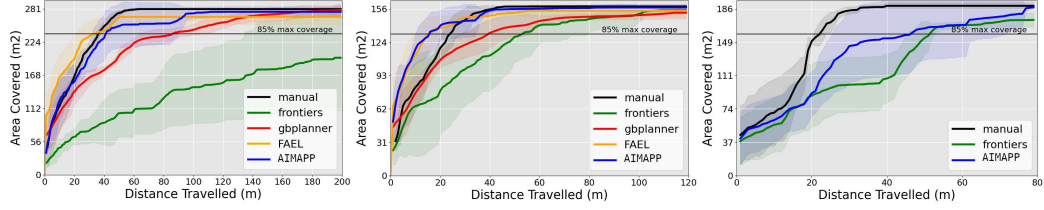
Across all scenarios, the agent begins each trial from a different initial position. Exploration results comprise five successful trials per environment per model. Failures and human interventions are documented in Appendix D.3. Drift measurements are conducted over five runs per model in the 185 m² warehouse using Qualisys cameras, exclusively for benchmarking (no navigation model has access to the ground-truth positioning). Goal-reaching is evaluated over 40 runs across all environments (simulated and real) with goals at various distances from the starting position; there is 17 goal-oriented real-world trials across three environments. In those goal-reaching experiments, the agent is assumed to know its starting position, thereby bypassing the initial re-localisation steps. This isolates the goal-reaching behaviour from the re-localisation behaviour, since re-localisation requires several preliminary steps in a recognisable region before goal-directed planning can begin. Re-localisation under the same generative model has been evaluated separately in a simplified 2D MiniGrid setting without realistic observations [13]. Since our model does not construct a metric map, we associated LiDAR range measurements with internal state representations to evaluate spatial coverage between models.

4.2 Exploration

Simulation Results

The average coverage efficiency of each model in the two largest simulated environments, measured as explored area relative to distance travelled and averaged over five successful trials, is reported in Figure 7 and Table 2. Manual explorations are treated as a near-optimal reference, given that the human operator has a general understanding of the layout while navigating.

Across all simulated environments (including the additional environments in Appendix D.2, Table D3), AIMAPP achieves a coverage efficiency of 90.3% relative to manual exploration. In the large warehouse, AIMAPP attains a CE comparable to FAEL and substantially above GBPlanner and Frontiers. In the house environment, AIMAPP and FAEL both exceed manual performance, while GBPlanner and Frontiers lag. The manual navigation is outperformed due to the humans conscientiously visiting all areas while the models accept ignoring some small unknown parcels and focus on bigger unknown areas, leaving those small unvisited parcels for later. The nAUC values follow a similar pattern: AIMAPP and FAEL achieve the highest normalised coverage progression, while GBPlanner and Frontiers are consistently lower.



(a) Large warehouse (b) Home (c) Real warehouse
Fig. 7 Coverage efficiency considering the agent’s travelling distance: (a) large simulated warehouse, (b) simulated home, and (c) real warehouse (Frontiers and AIMAPP only).

A coverage plateau at approximately 90% is observed in AIMAPP. This arises because the model prioritises updating nearby unvisited states already represented in its internal graph rather than extending exploration beyond the current detection range, resulting in node refinements without proportional increases in spatial coverage. In smaller environments ($\leq 80 \text{ m}^2$), all methods converge to similar performance levels (see Appendix D.2).

Frontiers performed worst overall, as it repeatedly attempts to reach unreachable frontier cells, wasting significant travel distance. The GBPlanner method prioritises safe exploration paths, often performing redundant back-and-forth movements before expanding to new zones.

Table 2 Exploration efficiency metrics across environments.
 CE: Coverage Efficiency (m^2/m), nAUC: Normalised Area Under Coverage Curve. Values are mean $\pm$ std over 5 runs.

Env.	Model	CE	nAUC
Simulated Large Warehouse	Manual	4.25 ± 0.43	0.55 ± 0.02
	AIMAPP	4.02 ± 1.24	0.62 ± 0.06
	Frontiers	0.60 ± 0.35	0.53 ± 0.15
	FAEL	4.65 ± 0.45	0.77 ± 0.07
	GBPlanner	1.13 ± 0.17	0.56 ± 0.04
Simulated House	Manual	4.02 ± 0.74	0.70 ± 0.08
	AIMAPP	4.89 ± 1.10	0.87 ± 0.10
	Frontiers	2.18 ± 0.15	0.77 ± 0.17
	FAEL	5.23 ± 1.90	0.79 ± 0.06
	GBPlanner	3.70 ± 0.15	0.50 ± 0.04
Real Warehouse	Manual	4.89 ± 0.53	0.65 ± 0.09
	AIMAPP	2.68 ± 0.40	0.70 ± 0.10
	Frontiers	2.11 ± 0.52	0.62 ± 0.15

Real-world Results

In real-world trials, only AIMAPP and Frontiers could be deployed, as FAEL and GBPlanner require sensors unavailable on the physical platform. Both were tested from five different starting positions in the warehouse environment, which contained two long aisles and a large open area. Results are reported in Figure 7(c) and Table 2.

AIMAPP was also tested in two additional environments, a small bedroom (20.3m²) and a dynamic parking lot (325 m²), in which Frontiers consistently failed before completing the trial owing to excessive odometric drift on wooden and sandy flooring; failure modes are documented in Appendix D.3.

AIMAPP consistently achieves faster coverage than Frontiers. The difference stems from strategy: AIMAPP moves efficiently between unexplored regions, while Frontiers repeatedly revisits the same aisles due to attraction toward unreachable frontier cells without planning through alternative paths. Performance varies with initial placement, reflecting the influence of warehouse geometry.

Frontiers performed comparatively better in the real warehouse than in the simulated house of similar dimensions, likely because long aisles are more forgiving for its greedy strategy. Conversely, AIMAPP performed slightly less efficiently in the real warehouse than in simulation, as LiDAR detection errors occasionally caused it to attempt reaching unreachable goals (behind an aisle), introducing inefficiencies in motion planning. Exploration results for the parking lot and bedroom are provided in Appendix D.2. In the parking, navigation was paused whenever a car moved through the space; we did not evaluate the obstacle avoidance performance of the motion planner, as it is not part of the proposed model.

Success Rates

Table 3 reports the success rate for obtaining five successful exploration trials per environment. GBPlanner achieved the highest reliability (87%), followed by AIMAPP (79%), Frontiers (52%), and FAEL (48%). Failures in AIMAPP occurred primarily when localisation drift in large unexplored areas produced a non-recoverable map overlap, or when the robot flipped after colliding with an obstacle undetected by the LiDAR. GBPlanner failed only when undetected obstacles caused the robot to flip. Frontiers frequently stalled by persisting in attempts to reach unreachable goals, while FAEL exhibited failures due to mismatches between topological node creation and obstacle detection in its 3D mapping. FAEL and GBPlanner were not evaluated in real-world settings due to sensor requirements incompatible with the available hardware. Detailed intervention logs and per-environment breakdowns are provided in Appendix D.3.

Table 3 Percentage of exploration success rate for each model over each environment.

External intervention	AIMAPP	FAEL	Gbplanner	Frontiers
Home	0.71	0.45	0.83	0.63
Big warehouse	1	0.5	1	0.83
Small warehouse	1	0.63	0.87	0.83
Mini warehouse	0.83	0.83	1	0.83
Real home	0.6	x	x	0.09
Real parking	0.75	x	x	0.00
Real warehouse	0.63	x	x	0.45

4.3 Robustness

Obstacle Adaptation

AIMAPP dynamically adapts to environmental changes during navigation by continuously updating its internal topological map. This process is illustrated in Figure 8 in a small-scale (25 m^2) environment where the position of a box is displaced from $(-1, 0)$ to $(0, -1)$ during exploration. As the agent moves and attempts transitions, it incrementally weakens the likelihood of inaccessible paths and reinforces the plausibility of reachable ones. Failed attempts to move toward an obstructed location trigger significant updates to the agent’s belief structure, while successful transitions reinforce existing connections.

In Figure 4.3, state 8 (position $(0, -1)$) becomes inaccessible after being blocked by the displaced box. All transitions leading to it are suppressed as the agent gathers evidence by navigating around the obstacle over approximately twenty steps. Numerical details regarding the belief update mechanism and associated learning rates are provided in Appendix A.1.

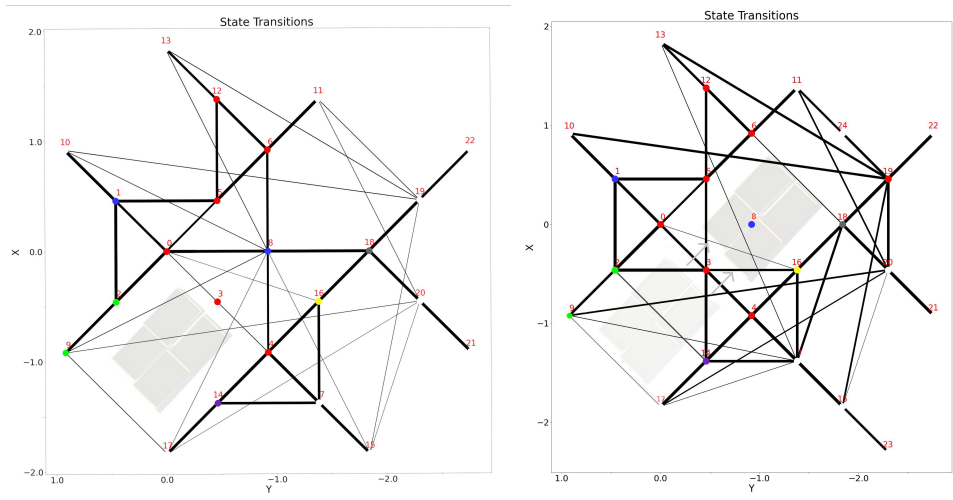


Fig. 8 Adaptation to a displaced obstacle: a box was moved from (a) $(-1, 0)$ to (b) $(0, -1)$ during exploration of a 25 m^2 environment. Over twenty steps around the box, the map weakens impossible transitions and reinforces previously improbable links. New nodes are created if pertinent.

Drift Measurement

To quantify drift under controlled conditions, we conduct experiments in the 185 m^2 warehouse equipped with Qualisys motion-capture cameras providing ground-truth odometry. These cameras are exclusively used for benchmarking and are not available to any navigation model. The trajectories estimated by AIMAPP (believed odom), the robot’s onboard sensor odometry (robot odom), and the ground truth (real odom) are qualitatively compared in Figure 9. Missing segments correspond to gaps in Qualisys coverage.

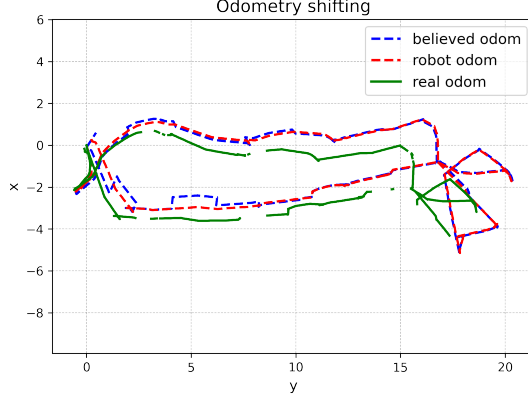


Fig. 9 Drift comparison in the (x, y) plane between AIMAPP’s internal belief (model odometry), the robot’s onboard sensor odometry, and the ground-truth trajectory measured with Qualisys. Missing segments correspond to gaps in ground-truth coverage.

Table 4 reports both the absolute RMSE on the (x, y) axes and the trajectory-length-normalised drift (ATE/m), averaged across five successful runs. In absolute terms, AIMAPP and Frontiers achieves comparable mean RMSE, but AIMAPP exhibits substantially lower variance, indicating consistently reliable trajectories where Frontiers occasionally diverges. The manual baseline was obtained over shorter trajectories at operator-chosen speeds, providing less opportunity for absolute drift to accumulate.

Table 4 RMSE over x and y , and ATE/m, averaged over 5 runs per model.

	AIMAPP		Frontiers	Manual
	model	sensor	sensor	sensor
RMSE (x, y)	1.83 ± 0.77	1.65 ± 0.79	1.68 ± 2.00	1.40 ± 0.25
ATE/m	0.20 ± 0.10	0.21 ± 0.10	0.58 ± 0.17	0.06 ± 0.02

Once the trajectory length is factored out, AIMAPP achieves substantially lower per-metre drift than Frontiers (ATE/m), approximately three times lower. We attribute Frontiers’ higher per-metre drift to its greedy exploration strategy: repeated attempts to reach unreachable frontier cells and redundant backtracking accumulate odometric error without proportional spatial progress, inflating the per-metre error.

The near-equivalence of the AIMAPP model and sensor rows confirms that the belief layer tracks sensor odometry closely on the metric trajectory. It does not continuously correct drift, but corrects it opportunistically when stored panoramic observations are recognised. In the case of exploration, this mechanism does not occur when navigating in previously unexplored locations. The manual baseline achieves the lowest per-metre drift, reflecting smooth, efficient trajectories with minimal redundant motion.

AIMAPP’s drift advantage, therefore, combines two properties: lower per-metre error than the autonomous baseline (Frontiers), and lower variance in absolute RMSE,

with no catastrophic failures. On surfaces where Nav2’s occupancy mapping becomes unusable (sandy or carpeted flooring), AIMAPP continues operating while Frontiers fails outright.

Sensor Failure and Severe Drift

A more pronounced distinction emerges in environments prone to severe drift. In the parking lot (uneven, sandy terrain) and the bedroom (mixed wood and carpet flooring), wheel slippage caused large odometry errors that could not be quantified. Under these conditions, AIMAPP’s belief-driven map gradually lost metric alignment with the ground truth, yet the agent remained operational: it could continue exploring and reliably reach goal observations, provided that the goal was not estimated to lie beyond a wall. The system also tolerates temporary sensor failures; when odometry, camera, or LiDAR streams were restarted after a failure, the agent resumed operation without requiring reinitialisation.

By contrast, Frontier-based exploration, which relies on Nav2, fails under these conditions: once drift is accumulated, its occupancy maps become inconsistent and unusable, and restarting the odometry is not an option for recovery. Consequently, no quantitative comparison with Frontiers was possible in the bedroom or parking lot scenarios.

4.4 Goal-Reaching

Goal-Directed Navigation Strategy

In our framework, goal-directed navigation emerges from the same Active Inference principles that drive exploration. While exploration favours previously unobserved states for their potential information gain, goal pursuit biases the agent toward states likely to generate a desired observation provided as input by the user. In these experiments, visual observations seen during exploration were given as the preferred observation C_o , with a weight of 10 on the pragmatic value to encourage the agent toward the goal over further exploration. All tests were conducted in partially explored environments where the goal had been observed during the prior exploration phase, to isolate goal-reaching behaviour from exploration.

This preference only influences behaviour if the agent can imagine a feasible trajectory toward the relevant states during planning. Far-removed high-utility states that do not appear in the planning horizon of the MCTS would not guide navigation. To address this, we employ an inductive prior H (formally introduced in Equations (4) and (A2)), which propagates utility along sequences of plausible actions leading to the goal. This mechanism allows the agent to evaluate paths rather than isolated states, effectively assigning intermediate states a utility proportional to their potential to reach the target observation, even outside its planning horizon.

This process is illustrated in Figure 10 in the simulated large warehouse. Heatmaps show the evolving utility of paths over five planning steps, with deeper red showing higher attractiveness. Initially (step 0), the agent (light green circle) is still biased in the correct general direction despite the goal states (circled in neon green) not yet being directly imagined from the current position. Two adjacent states correspond to the desired observation given to the agent; thus, there are two circled nodes. The executed

path in light green is shown over all steps to help visualisation over time. As the agent transitions through steps 1 to 4, intermediate states leading toward the goal increase in utility, while states in the opposite direction are progressively suppressed. At each step, the agent simultaneously updates its positional belief through re-localisation and evaluates future trajectories, maintaining flexibility in response to newly encountered information.

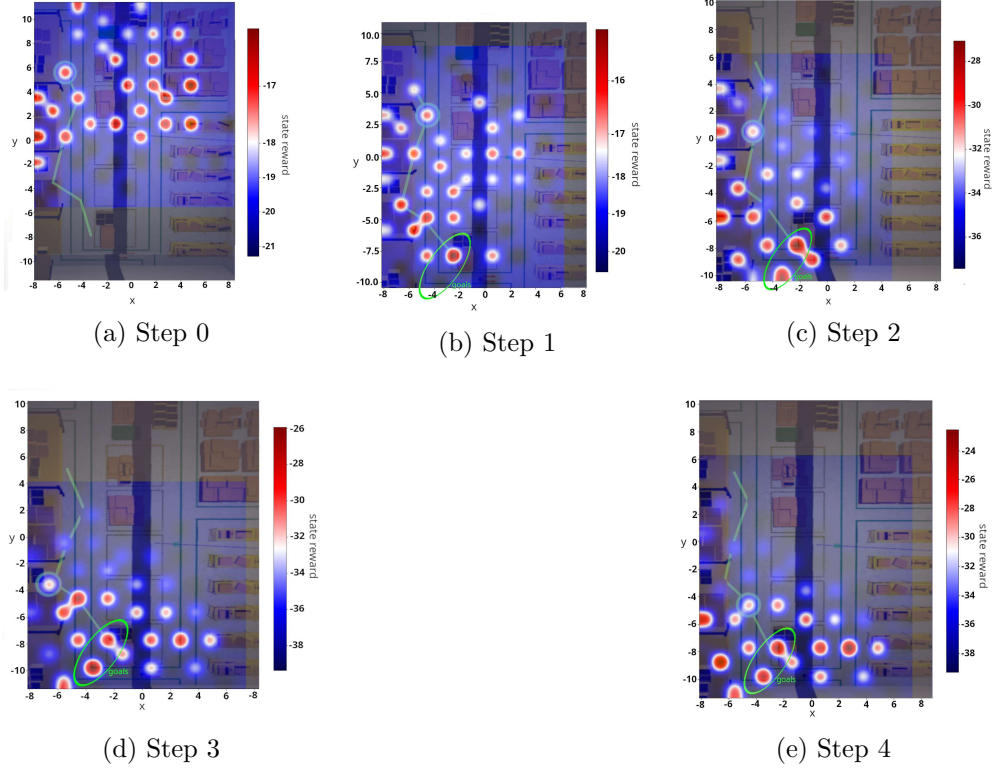


Fig. 10 Utility heatmaps over the agent’s topological map during MCTS planning toward a goal observation held by two states (circled neon green). Higher values indicate more attractive locations. The agent reached the goal in five steps from the starting pose (circled light green in step 0). The full trajectory is shown in dark green, with the current position circled. In step 0, the agent is attracted in the correct direction before it can directly imagine the goal state.

Quantitative Goal-Reaching Performance

To quantitatively assess goal-directed navigation, we compared the executed path length against two baselines: 1) the ideal path through AIMAPP’s topological map as constructed during prior exploration, and 2) an A* path computed over the occupancy grid of each environment. We report path lengths in metres and their three-way decomposition as Path Length Ratios (PLR) in Table 5. The executed path against A*

on a complete occupancy map, and against AIMAPP’s ideal transition planning (as we would expect the model to behave). Moreover, AIMAPP’s ideal transition plan is compared to A* to analyse how good AIMAPP can be, at its best. In all goal-reaching runs, the agent was assumed to know its starting position to bypass re-localisation, isolating goal-reaching from re-localisation behaviour.

Table 5 Goal-reaching path efficiency across 40 runs. PLR: Path Length Ratio (executed / reference).

Comparison	PLR (mean $\pm$ std)	Median
Executed vs. A* (true graph)	1.260 ± 0.202	1.240
Executed vs. AIMAPP plan	1.163 ± 0.160	1.186
AIMAPP plan vs. A* (agent graph)	1.085 ± 0.111	1.049

We conducted 40 goal-reaching runs across all environments with goals at varying distances from the starting position. The end-to-end PLR decomposes approximately multiplicatively: $1.260 \approx 1.085 \times 1.163$. Most of the inefficiency arises when the agent encounters transitions previously believed to be feasible but actually blocked by an obstacle, and must replan. Figure 11(a) shows the executed distance against the ideal travelled distance (in AIMAPP topological graph) for each run, and Figure 11(b) shows the distribution of PLR across runs. The agent reached the goal within 20% of the A* optimum in 38% of trials, within 50% in 88% of trials, and exceeded 50% in 12% of trials, with a median PLR of 1.24.

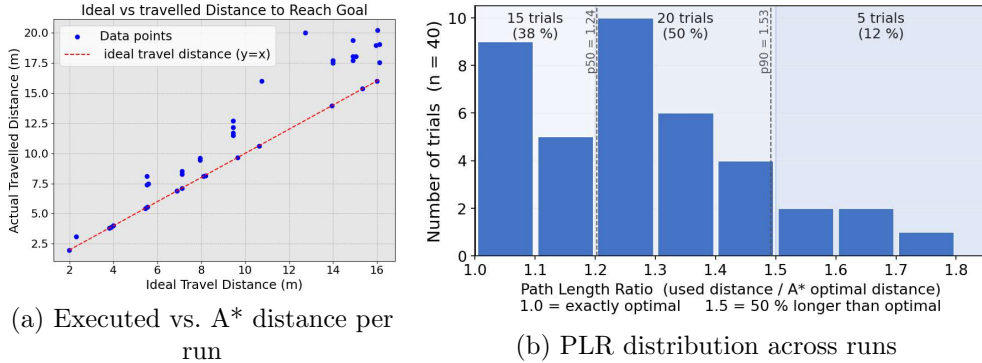


Fig. 11 Goal-reaching path efficiency over 40 runs across all environments. (a) Executed distance against ideal AIMAPP distance per run. (b) Distribution of the Path Length Ratio (executed / A*); 38% of trials lie within 20% of A*, 50% between 20% and 50% over A*, and 12% over 50%, with median PLR = 1.24.

Performance is not uniform across goals. One goal in the simulated large warehouse consistently produced $PLR \approx 1.50$ across five runs with low variance (± 0.27), accounting for three of the five worst-performing trials in the dataset. The A* distance for this goal is short (4.53 m) while the agent’s executed distance is 7.4 – 8.1 m, a near-doubling that we attribute to a conservative belief in B_p : the learned transition

model assigns low feasibility to a direct path that A* on the true graph traverses freely. The remaining failures are distributed across distinct goals, suggesting that systematic detour behaviour is goal-specific rather than environment-specific. An example of such a case can be seen in Figure 12.

Figure 12(c) shows three executed paths from a fixed starting location to the same goal in the simulated warehouse, alongside the ideal trajectory (dashed black). Across trials, the agent reached the target observation while prioritising transitions with high expected utility; slightly longer paths are preferred when they offer higher confidence in expected outcomes, consistent with the Active Inference principle that action selection balances efficiency with belief consistency.

In real-world trials, the agent was evaluated in three environments of varying complexity: the controlled bedroom (2 runs), the structured warehouse (3 runs) and the semi-structured parking lot (12 runs). Figure 12(d) illustrates trajectories taken in the parking lot to reach the corresponding goal image (b). Despite sensor drift, environmental change between exploration and goal-reaching phases (cars moving through the parking space) and varying lighting conditions (light on or off), the agent successfully reached its goals in 15 of 17 real-world runs. When initial localisation errors or unexpected obstacles occurred, belief updates and replanning enabled the agent to correct its course, as illustrated by minor detours in the green trajectory, where the agent got temporarily lost after drifting over an undetected obstacle (a car with a body higher than its lidar range).

Across the seventeen real-world goal-reaching experiments, only two failures occurred: one in the bedroom, caused by severe drift producing an incorrect re-localisation with the goal located beyond a wall, and one in the parking lot, where the robot’s holonomic wheels became physically stuck in sandy terrain.

4.5 Computational Scalability

To evaluate the scalability of AIMAPP, we measured its computational footprint in terms of memory usage and runtime performance on a Jetson Orin Nano platform (15V power, Jetpack 6.1, Ubuntu 22.04, ROS2 Humble).

Model Size and Memory Usage

Model size was assessed by serialising parameters into a .pk1 file across 31 independent runs. The number of stored states had no measurable effect on model size, confirming that the state-space representation remains lightweight as the map grows. The dominant memory factor is the storage of RGB panoramic observations. Model size (in MB) scales linearly with the number of stored observations:

$$\text{Model size (MB)} \approx 1.32 \times (\text{number of observations}) + 0.17.$$

The largest model contained 36 unique observations and required only 44.9 MB.

Runtime and Resource Usage

Runtime performance was profiled over 31 navigation runs on the Jetson Orin Nano. During each run, AIMAPP operated concurrently with the Nav2 stack, camera

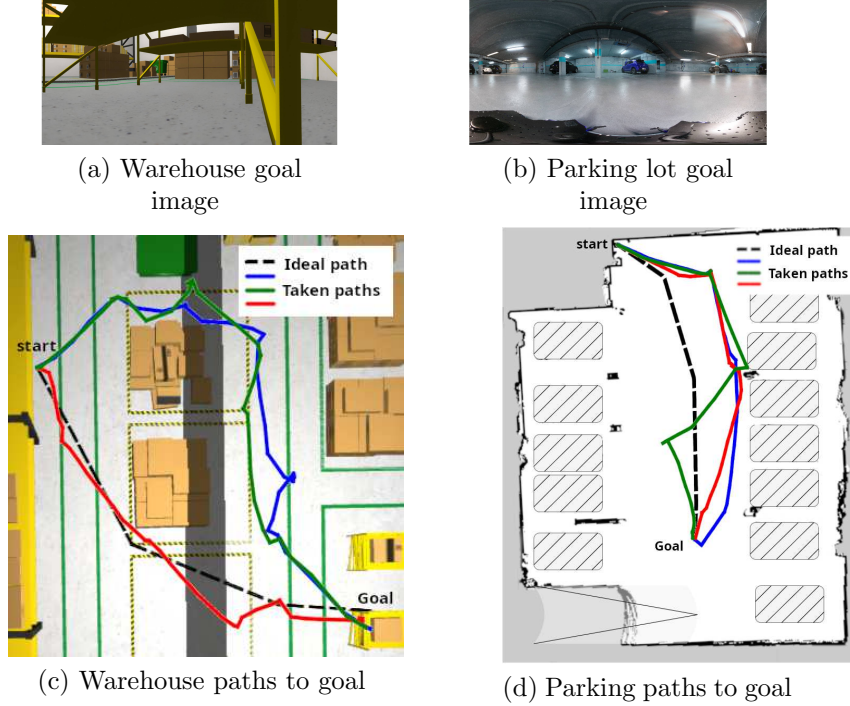


Fig. 12 Goal images and example paths taken by the agent (red, green, blue) from the start to the goal, alongside the ideal trajectory (dashed black line), in the simulated large warehouse (a, c) and the real parking lot (b, d).

drivers, plotting, and logging processes. The average system resource consumption was $47.5\% \pm 16.8\%$ CPU and $36.8\% \pm 2.2\%$ RAM. Nav2 accounted for approximately 30% of CPU usage and remained stable across all trials. At no point did computation exceed platform capacity or cause performance degradation. As shown in Figure 4.5, CPU load varied dynamically with task demands (periodic data and plots saving requiring more resources) but remained within approximately 60% of available capacity. For comparison, the Frontiers baseline exhibited $42.7 \pm 26.2\%$ CPU and $26.3 \pm 1.2\%$ RAM usage over 14 runs.

Execution Time and Model Dimensionality

Model execution time scaled linearly with model dimensionality (the number of states considered at each planning step):

$$\text{Execution time (s)} \approx 0.18 \times \text{Model dimension} - 2.18.$$

Figure 4.5 illustrates this relationship. The use of MCTS for policy evaluation introduces computational overhead that scales with planning depth and simulation count. In our implementation, we used depth 10 and 30 simulations, chosen empirically as sufficient for the tested environments. Preliminary experiments in small environments suggest these could be reduced to depth 5 and 20 simulations with minimal

performance degradation, improving real-time performance on resource-constrained platforms.

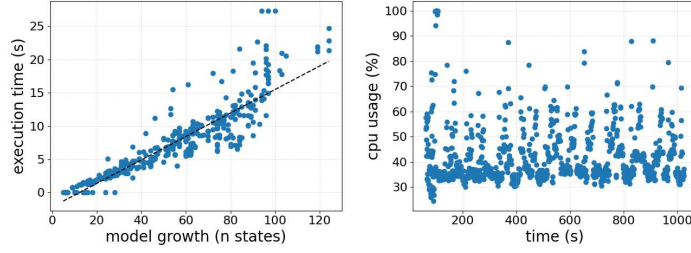


Fig. 13 Impact of AIMAPP with MCTS policy evaluation and Nav2 on the Jetson Orin Nano, measured over 31 experiments.

5 Discussion

The presented results demonstrate that AIMAPP achieves competitive exploration and goal-directed navigation performance, comparable to state-of-the-art planning methods that do not require pre-training, in both simulated and real-world environments up to 325 m². We now discuss the limitations of the current system, situate it within a broader context relative to existing approaches, and identify directions for future work.

5.1 Limitations

Planning and State Representation

The current implementation relies on discretised actions (x headings) and a fixed node influence radius, which may limit performance in highly constrained environments such as narrow passages where finer spatial resolution is required. Autonomous adaptive discretisation, reducing the influence radius or increasing angular resolution in cluttered regions, could address this limitation.

The discrete-state representation does not account for the uneven distribution of information across space. A large open hall may be over-represented in the topological graph despite containing little novel information, while a narrow alley may provide greater utility but be collapsed into a few nodes. This inefficiency becomes more pronounced in larger environments.

The coverage plateau observed at approximately 90% (Section 4.2) reflects a related limitation: the agent prioritises refining nearby unvisited states already in its graph rather than extending exploration beyond the current detection range. While this conservative strategy contributes to robustness, it reduces the rate at which new spatial coverage is acquired in the later stages of exploration.

Localisation and Drift

Perceptual aliasing in symmetric environments is mitigated by the position variable p_t , which accumulates metric displacement over time. While identical visual observations would yield equal SSIM scores, the accompanying position estimates enable disambiguation. However, this mechanism depends on continuous tracking; if the robot were teleported (kidnapped robot problem), it would be unable to distinguish between symmetric locations and would likely converge to an incorrect belief with probability proportional to the number of indistinguishable locations. In our experiments, repeated visual patterns in corridor environments were successfully handled through this position-based disambiguation.

We investigated marginal message passing considering observations and actions up to four steps in a previous work over a simple 2D minigrid environment without realistic features [13] as a form of belief smoothing for loop closure. However, this retrospective inference increased uncertainty over the current state rather than improving localisation, yielding performance below the single-step model we are currently using.

Under extreme drift conditions, the belief-driven map gradually loses metric alignment with the ground truth. While the agent remained operational in these conditions, the two goal-reaching failures in real-world trials were both attributable to severe drift: one produced an incorrect re-localisation with the goal located beyond a wall, and the other resulted from the robot becoming physically stuck. Long-term metric consistency remains an open challenge for industrial deployment. Integrating a geometric back-end behind the topological layer, so that metric drift is corrected between re-localisation events rather than only at them, is a natural direction for closing this gap.

Perception

The perception front-end is not part of the proposed Active Inference model; it is an interchangeable module whose current implementation uses panoramic images compared via SSIM as a proof-of-concept. This choice introduces several weaknesses that are specific to the perception pipeline rather than to the decision-making framework.

First, the SSIM-based pipeline is brittle to environmental change, i.e., small modifications in the scene (e.g., a displaced object or altered lighting) can prevent the agent from recognising the goal upon reaching the correct location. This brittleness was directly responsible for reduced goal recognition reliability in real-world trials, where environmental conditions varied between exploration and goal-reaching phases. In the limiting case, a goal location defined by a panoramic image may become entirely invalid if the scene has changed sufficiently that no stored observation matches the current input, even at the correct position. Replacing SSIM with semantic object detection or learned visual representations would address this without modifying the generative model.

Second, the observation likelihood model A_o associates multiple observations with each state, providing a mechanism for capturing temporal variability. However, as the map expands, observation likelihood distributions for earlier states naturally dilute, potentially requiring the agent to revisit previously mapped locations to refresh perceptual beliefs. This dilution reflects an inherent trade-off in dynamically updated

cognitive maps compared to static representations, and is independent of the particular observation modality.

Third, the current evaluation does not include continuously dynamic environments such as spaces with pedestrian traffic. While the framework accommodates moderate environmental changes (as demonstrated with relocated obstacles in simulation), performance in settings with frequent structural change remains untested. Handling such scenarios requires robust dynamic object filtering at the perception level and reactive obstacle avoidance at the motion-planning level, neither of which falls within the scope of this work. Finally, the current implementation assumes approximate heading consistency between visits to the same location. The 360° panoramic coverage provides sufficient overlap to tolerate moderate heading variation, but a better perception module invariant to this kind of consideration would be beneficial.

5.2 Broader Context

Comparison with Baselines

The baselines evaluated in this work reflect distinct design philosophies. AIMAPP and FAEL both emphasise exploration efficiency, achieving the highest coverage-to-distance ratios. However, FAEL requires a 3D LiDAR and achieves a success rate below 50%, frequently stalling due to mismatches between topological node creation and obstacle detection in its 3D mapping. GBPlanner is tailored for robust subterranean exploration where narrow passages dominate; it achieves the highest success rate (87%) but at the cost of conservative, distance-inefficient path planning with redundant back-and-forth movements in large environments. Frontiers, as a lightweight heuristic, performed worst in cluttered environments due to its greedy, reactive strategy.

A distinguishing property of AIMAPP is that it unifies exploration and goal-directed navigation within a single generative model, without requiring retraining, reinitialisation, or pipeline restructuring. Unlike FAEL, GBPlanner, and Frontiers, which are explicitly optimised for exploration alone, AIMAPP switches between discovering new areas and reaching specific observations by adjusting hyperparameters controlling the balance between pragmatic value and information gain. The framework naturally supports both objectives simultaneously.

Comparison with Online Learning Approaches

Comparison with other online learning approaches represents a valuable direction. Active NTFields [26] is the most relevant comparator for the online learning property of AIMAPP, as it also maps an environment from depth observations without pre-training and enables fast motion planning once the map is constructed. However, the two systems differ in problem scope, sensor requirements and computational demands, which together preclude a direct comparison under our current experimental setup. Active NTFields separates mapping from planning: it first explores the environment using a standard frontier-coverage strategy to build a complete neural arrival-time field. It does not support goal-directed navigation during partial exploration. AIMAPP, by contrast, interleaves exploration and goal-reaching within a single decision-making loop via EFE minimisation, operating on incomplete maps throughout. Comparing exploration coverage alone would not capture this difference. On the

sensor side, Active NTFields requires dense 3D depth images with accurate pose registration and GPU-accelerated neural network training (reported on a 3090 RTX with 128 GB RAM and 2.12 seconds per frame). AIMAPP operates with a 2D LiDAR and a forward-facing RGB camera on a Jetson Orin Nano. Evaluating their relative exploration efficiency would provide further insight into the trade-offs between physics-informed neural representations and topological generative models, which we leave to future work.

Relation to Cognitive Mapping

The stepwise propagation of utility observed in the goal-reaching experiments (Figure 10) resembles place cell firing in the hippocampus, where spatial locations associated with expected sensory outcomes are activated before the agent physically reaches them [54]. Forward sweeps of place cell activation have been observed as organisms consider alternative routes, with the most robust sweeps predicting the direction of subsequent locomotion [55, 56]. This is consistent with the view of place cells as encoding a "predictive map" based on successor representations of likely state transitions [57], from which anticipated future locations are derived before the agent arrives at them. We do not claim that AIMAPP implements these human mechanisms, only that the inductive prior produces a comparable forward-projection pattern from a probabilistic inference.

By integrating predicted observations with trajectory planning, AIMAPP dynamically selects actions maximising the likelihood of reaching desired sensory states while accounting for uncertainty. This parallel suggests that our AIF application and biological navigation may share structural features through the inductive prior mechanism and hippocampal predictive coding. The current perception pipeline, using panoramic images and SSIM-based comparison, could be replaced by learned visual representations (e.g., from a pre-trained encoder or a visual place recognition network) without modifying the underlying Active Inference framework. Such an extension would more closely parallel the hierarchical processing from visual cortex to hippocampus observed in biological systems, where abstract, viewpoint-invariant representations are formed before reaching place-coding regions [58]. The modularity of AIMAPP explicitly supports this separation: the generative model operates on abstract state and observation variables, remaining agnostic to how those observations are derived from raw sensory input.

Sensor Agnosticism

While this work uses LiDAR to detect obstacles through the observation vector o , the model treats depth as a generic perceptual feature. Future implementations could replace LiDAR with depth-from-vision or any other sensing modality without altering the underlying decision-making process. This sensor-agnostic design, combined with the ROS-based integration, ensures that AIMAPP can be deployed across diverse robotic platforms with minimal adaptation.

5.3 Future Work

Several directions could address the limitations identified above and extend the capabilities of the framework.

Perception Improvements

The SSIM-based observation comparison should be replaced by a more robust recognition module. Semantic object detection [59, 60] would improve goal recognition, particularly for task-critical observations where small environmental changes currently prevent the agent from confirming arrival at the correct location. More broadly, the panoramic image pipeline could be substituted with learned visual representations from a pre-trained encoder or a visual place recognition network [58], which would provide viewpoint-invariant features better suited to real-world variability. Integrating dynamic object filtering would further improve performance in environments with moving entities. These replacements require no modification to the generative model itself, as the Active Inference framework operates on abstract observation variables.

Planning Improvements

A long-standing limitation of the current framework is the finite planning horizon imposed by MCTS, which restricts the depth over which the agent can evaluate future trajectories. Probability dilution over long planning horizons remains an open challenge in AIF. Scalable methods for balancing information gain and goal achievement in large state spaces with long planning horizons remain an open challenge in AIF [33, 61]. Hierarchical reasoning [37] offers the most principled mitigation: by separating long-term topological planning from short-term metric control, a hierarchical generative model would enable the agent to reason over coarser spatial abstractions at higher levels while preserving fine-grained resolution at lower levels [37, 62]. This would simultaneously extend the effective planning horizon and accelerate coverage in large open areas. Complementarily, grouping states by information gain [63–65] would reduce redundancy in the topological graph by collapsing over-represented regions (e.g., open halls) into fewer abstract nodes, improving both map compactness and path planning efficiency. Within such a hierarchy, message-passing schemes [66] could propagate beliefs across temporal scales, enabling the agent to reason about distant goals that currently fall outside its prediction horizon without sacrificing the computational tractability of the single-level model.

At the level of the current flat model, more sophisticated MCTS rollout policies incorporating learned heuristics (e.g., A* cost estimates) could improve planning efficiency as the map grows; however, this direction remains untested in the present implementation.

Environmental Robustness

Adaptive discretisation, autonomously reducing the influence radius or increasing angular resolution in cluttered regions, would improve performance in narrow passages. Continuous action spaces [11] offer a theoretically principled extension but require alternative inference schemes, such as gradient-based optimisation, and are left for future investigation.

Alternative temporal smoothing architectures should be explored to revisit belief smoothing for loop closure.

Finally, while the current system tolerates temporary sensor failures by pausing until data becomes available, developing graceful degradation strategies for prolonged sensor loss would be essential for deployment in safety-critical settings.

6 Conclusion

We have presented AIMAPP, an Active Inference-based navigation framework unifying mapping, localisation, exploration, and goal-directed navigation within a single generative model. The agent builds and updates a sparse topological map online, learns state transitions through sensorimotor feedback, and plans actions by minimising Expected Free Energy, requiring no pre-training, metric maps, or task-specific tuning.

Across seven environments (four simulated, three real-world, up to $325m^2$), AIMAPP achieved 90% of human-teleoperated coverage efficiency in simulated environments, and 79% exploration success rate overall. In goal-reaching experiments, AIMAPP achieved a median path length 24% above the A* optimum across 40 trials, with 15 of 17 real-world trials successful. These results provide evidence that AIF can serve as a practical navigation strategy for mobile robots, bridging neuroscience-inspired models of spatial cognition and the demands of real-world autonomy. The modular, sensor-agnostic architecture of AIMAPP supports integration with existing ROS-based systems and permits straightforward substitution of perception, planning, or control modules as they improve. Limitations in goal recognition, planning horizon, and long-term metric consistency are future research areas with plausible solutions, including hierarchical reasoning, semantic perception, and adaptive state discretisation.

Data. Our model and videos are available at `//Commented for anonymity//`

Acknowledgements. Large language models were used to assist with grammar correction and linguistic articulation during the preparation of this manuscript.

[Commented for anonymity]

Declarations

Some journals require declarations to be submitted in a standardised format. Please check the Instructions for Authors of the journal to which you are submitting to see if you need to complete this section. If yes, your manuscript must contain the following sections under the heading ‘Declarations’:

- Funding Check
- Conflict of interest/Competing interests (check journal-specific guidelines for which heading to use) Not applicable
- Ethics approval and consent to participate Not applicable
- Consent for publication Check
- Data availability Not provided
- Materials availability Check

- Code availability Check
- Author contribution Check

If any of the sections are not relevant to your manuscript, please include the heading and write ‘Not applicable’ for that section.

Appendix A Model Definition

A.1 Model Parameters

This section outlines the key parameters used to configure our active inference-based navigation model. These parameters define the agent’s internal representation, planning process, and learning dynamics. Hyperparameters set at initialisation are presented in Table A1. All values were chosen empirically without comparison testing.

A.1.1 Transition Update

$$B_\pi = B_\pi + Q(s_t|s_{t-1}, \pi)Q(s_{t-1}) * B_\pi * \lambda \quad (\text{A1})$$

To update its beliefs about the environment, the agent uses a Dirichlet-based pseudo-count mechanism (Equation A1) with situation-dependent learning rates (λ). These rates vary based on whether the agent:

- successfully reaches a location,
- is physically blocked while trying to move,
- or anticipates reaching (or failing to reach) a location based on sensory evidence.

Table A2 lists the specific learning rates for each situation. This continual adjustment allows the agent to refine transition likelihoods in its internal model rapidly.

A.1.2 Uncertainty About Current State

To determine whether an agent is lost, we evaluate its certainty about the current state. Specifically, we compute a Z-score to measure how strongly the most likely state stands out compared to the others. If this dominance falls below a user-defined threshold (set to 4 in this work), the agent is considered uncertain about its location.

Table A1 Hyperparameters of AIMAPP

Parameter	Description	Value Used
Number of likelihood matrices	Number of observation types (e.g., visual, pose). Theoretically extendable to LiDAR, radar, etc.	2 (visual and pose)
Number of available actions	Discretized actions over 360° rotation/movement space. Odd numbers include a "stay" action.	13 actions
Physical radius	Collision buffer for state creation near obstacles.	Robot-specific
Influence radius	Minimum distance between states to promote map compactness.	1 m (env. $\leq 40m^2$), 2 m (all others)
Lookahead nodes	Maximum consecutive nodes hypothesized within LiDAR range for anticipatory map expansion.	8 nodes
SSIM threshold	Below this threshold, two images are considered the same occurrence and the latest one overwrites the previous image.	65%
A_o Learning rate	How much new observations impact the observation Likelihood matrix update.	1
MCTS depth	How far can the simulation horizon predict (also called the lookahead horizon) - in step-	10
MCTS simulation runs	How many time do we run the MCTS simulation before determining which action is the most ideal	30

Table A2 Transition learning rate (λ) depending on the situation

Transitions	Possible	Impossible	Predicted Possible	Predicted Impossible
Forward	7	-7	5	-5
Reverse	5	-5	3	-3

A.2 Expected Free Energy Terms

H is an inductive term applied to the state preference Cs defined in Equation A2, which is computed inductively by propagating backwards in time from the goal state toward the current state (in n steps, that might be lower than the planning horizon). This inductive process accumulates structural preferences and becomes silent (i.e., has no influence) if no specific target state is preferred or if the preferred state lies beyond the predictive horizon. Let Cs_0 be a weighted preference vector over existing states, as derived from the joint preference over observations (Co) and transitions (Cp). Then the backwards recursion is defined as [45]:

$$\begin{aligned} Cs_n &= B_s^T \odot Cs_{n-1} \quad \text{for } n = 0, \dots, \tau \\ H &= \ln(\epsilon) \cdot (B_s^T \odot Cs_\tau) \odot s(\tau) \end{aligned} \quad (\text{A2})$$

n is a backwards value propagating from the goal (thus the future, if under our prediction horizon) up to our current state t , if Cs lies over the horizon, we can predict, this term will also be silent.

A.3 Monte Carlo Tree Search

We integrate Monte Carlo Tree Search (MCTS) with Active Inference to evaluate the expected free energy of potential paths, overcoming the limitations of fixed policy sets in standard Active Inference models [13, 30, 32]. Unlike traditional approaches, where policies are predefined and restrict adaptability, MCTS dynamically simulates future action sequences (rollouts) from the current belief state, guided by the principle of minimising expected free energy [34]. This enables deeper lookahead without combinatorial explosion, balancing goal-directed behaviour and exploration while reducing computational load. MCTS also allows the agent to revise trajectories based on new evidence, resulting in robust, context-sensitive navigation in uncertain environments. Our implementation (Algorithm 1) uses two key parameters: policy length (number of future steps to predict, set to 1) and lookahead policy (depth of simulation horizon, set to 10 in Equation A2). For each step, MCTS runs 30 simulations (number chosen empirically) to determine the optimal action.

Appendix B Robotic Platforms and Sensors

Our system is robot-agnostic; however, we have to adapt the sensor pipeline to the specific sensors used. All simulated experiments use a Turtlebot3 Waffle [67], except FAEL, which used a Jackal [68]. Our agent used a Pi camera and a 360-degree 2D

LiDAR with 12 m range, Gbplanner uses two 3D lidars and three Pi cameras and FAEL one 3D Lidar, all with a maximum range of 12m for fair comparisons. Real-world trials used a RosbotXL [69] with an 18 m range 360-degree LiDAR (restricted to 12 m for comparable results to simulated environments) and a 360-degree camera (Ricoh Theta X). In our implementation, the obstacle distance c was extracted from the LiDAR sensor rather than the visual observation o due to the absence of a depth or stereo camera; this substitution has no impact on the model, as the generative model treats depth as a generic perceptual feature.

Appendix C Environments

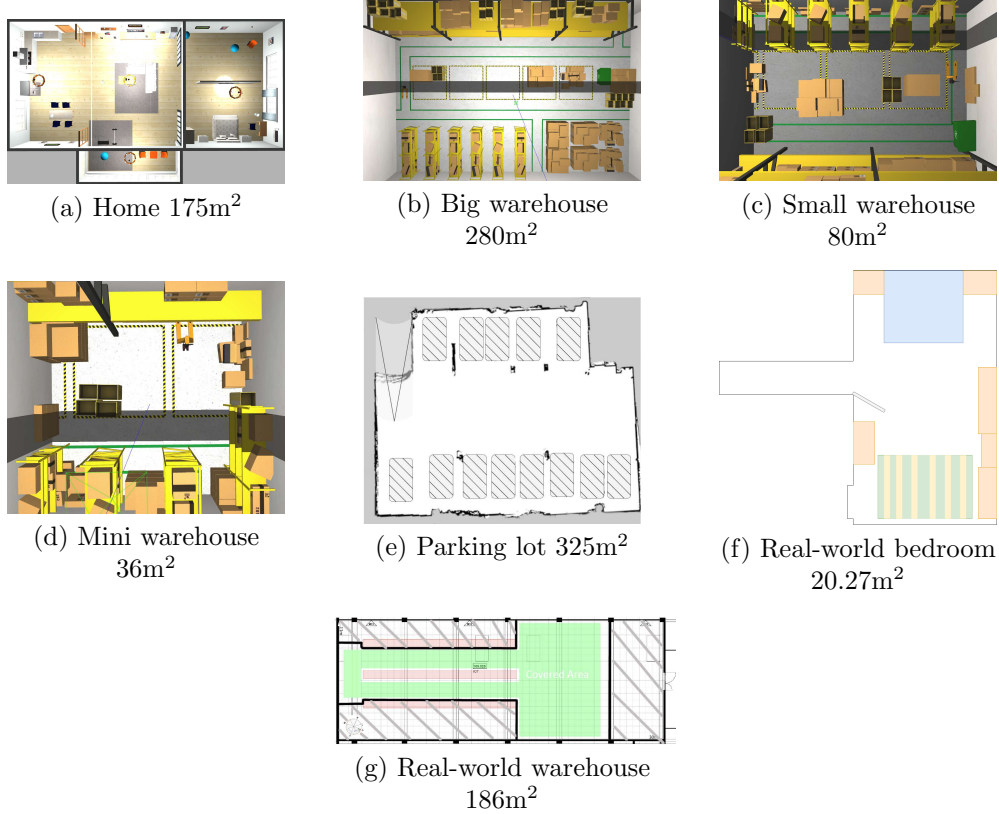


Fig. C1 Environments used in simulations and real-world experiments. Dashed grey areas are inaccessible; green areas correspond to the Qualysis odometry coverage.

Our experiments used a home [49] of 156m² and a warehouse [48] of 3 different sizes, ranging from 36m², 80m² up to 280² and a real-world house of 21m², a warehouse of 185m² and a parking lot of 325m². All environments are presented in Figure C1. Pictures are presented in Figure C2.



Fig. C2 Picture of the structured warehouse (left) and semi-structured parking lot (right)

Appendix D Details about experimental results

D.1 Adversarial Models

GBPlanner, FAEL, and Frontiers were used with their given parameters, except for the map resolution, which was increased to 0.05m/cell. Other parameters, such as obstacle inflation, were modified to ensure that all agents could physically reach every location.

Despite our efforts, FAEL wouldn't fully turn around the central box in the mini warehouse, explaining why Figure D3(b) coverage is not 100%.

D.2 Exploration paths

In smaller environments ($\leq 80m^2$), model performance was comparable across methods (Figure D3), matching the average of five manual explorations from identical start points (Table D3). This convergence stems from the limited LiDAR range, which rapidly covers most of the area and reduces the benefit of long-horizon planning.

Exploration trajectories for AIMAPP, FAEL, GBPlanner, and the Frontier-based approach are shown in Figure D3 (d-g). All agents started near the dumbbell, with trajectories progressing from black to yellow. FAEL, AIMAPP, and Frontiers reached nearly full coverage; GBPlanner achieved $\pm 95\%$, as it got trapped near the sports room chair.

Each strategy exhibited distinct behaviours. FAEL avoided the playroom but partially observed it from the doorway. Frontiers aggressively targeted unexplored regions, ensuring full coverage but with repeated backtracking and inefficient paths. AIMAPP alternated between forward progression and small corrective loops, accumulating more observations, but in this run, missed the playroom-bedroom connection on the far right. These qualitative differences are minor in small maps but grow significant in larger environments, where planning efficiency and robustness diverge more clearly.

Direct coverage comparisons between AIMAPP and FAEL or GBPlanner could not be performed in real-world settings due to incompatible sensor requirements. Moreover, Frontiers relied on Nav2's reliance on 2D occupancy maps, which are sensitive to drift and slippage. Consequently, results for Frontiers in the garage and home in the real world could not be fairly compared to our model. In these cases, drift and odometry loss led to misaligned or overlapping maps. To maintain consistency, AIMAPP's real-world coverage values were averaged over five runs and compared with a manual teleoperation using Nav2 SLAM. Figure D3(c) illustrates coverage in the parking

lot, where parked cars altered the free space between runs. Despite these variations, AIMAPP achieved consistent and robust coverage across multiple trials.

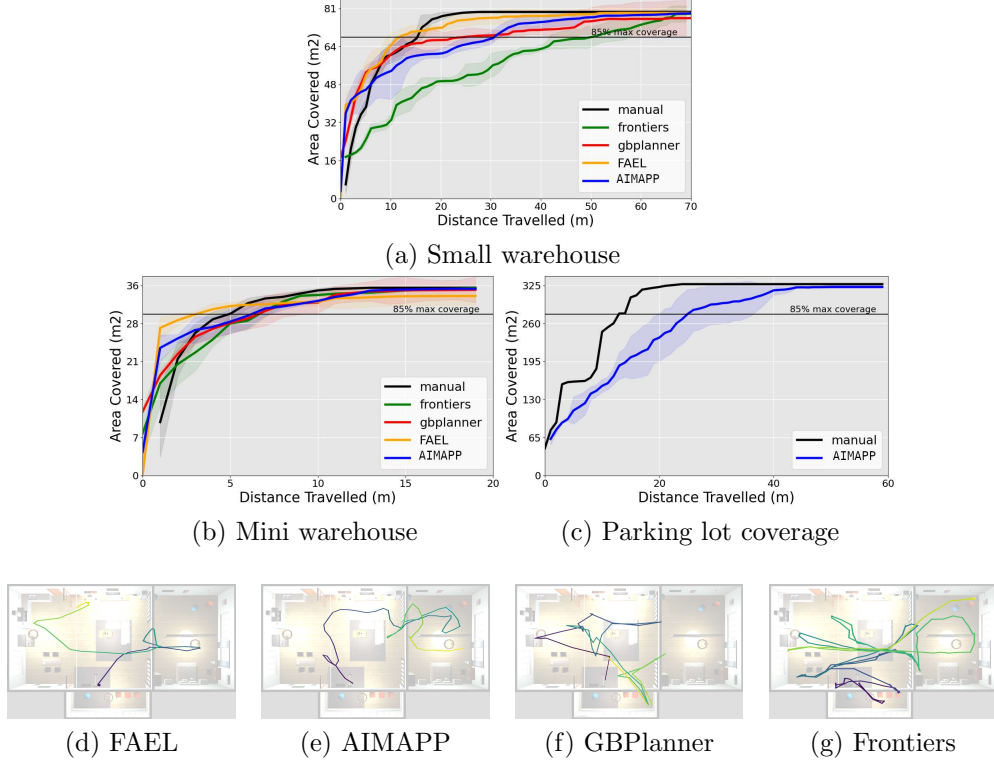


Fig. D3 Coverage efficiency in (a) small (80m^2), (b) mini warehouses (32m^2), and (c) real-world parking lot (325m^2). (d–g) Exploration trajectories for all models in the house environment; the shared starting point starts at the darkest end of the line and gradually shifts from black to yellow.

D.3 Reasons for human interventions

The reasons for human interventions varied substantially across models:

AIMAPP: Most interventions followed LiDAR blind spots (e.g., flat-based chairs, forklifts, or reflective walls). In these cases, the robot collided or spun in place, and the motion failure had to be signalled manually. Large drift in unexplored areas occasionally required manual re-localisation for benchmarking consistency.

Frontiers: Often persisted in unreachable goals, repeatedly trying to navigate toward them until manually repositioned.

FAEL: Failures arose when nodes were created near obstacles in Voxblox or when open-space detection misclassified traversable areas. Repositioning typically resolved issues, though the model sometimes refused to continue for unknown reasons.

GBPlanner: Showed the fewest failures but sometimes planned paths through partially detected obstacles, requiring manual adjustment. In AIMAPP, obstacle-related re-planning is delegated to the motion planner (Nav2), which reports motion failure; the decision layer merely selects goals. Interventions to prevent collisions with moving vehicles in the parking lot were excluded from Table [D4](#)

```

1 Algorithm: AIF based MCTS
   Input: possible_motions, policy_length, lookahead_policy, num_simulation
   Output: best policy
2 Function MCTS_planning(possible_motions, policy_length,
   lookahead_policy, num_simulation):
3   qs  $\leftarrow$  get believed localisation over states();
4   qo  $\leftarrow$  get expected observations for qs(qs);
5   root  $\leftarrow$  create root state(qs, qo, possible_motions);
6   for i  $\leftarrow$  1 to policy_length do
7     for i  $\leftarrow$  1 to num_simulation do
8       leaf  $\leftarrow$  Selection(root);
9       expanded_leaf  $\leftarrow$  Expansion(leaf);
10      reward  $\leftarrow$  Simulation(expanded_leaf, lookahead_policy);
11      Backpropagate(expanded_leaf, reward);
12      actioni  $\leftarrow$  SelectBestAction(root);
13      root  $\leftarrow$  leaf;
14      selected actions  $\leftarrow$  actioni
15    return selected actions;
16 Function Selection(node):
17   while node has children do
18     node  $\leftarrow$  child of node with highest UCB1 score;
19   return node ;
20 Function Expansion(node):
21   foreach action in node.possible_actions do
22     next_pose  $\leftarrow$  Transition(node.pose, action);
23     next_pose, qs', qo', G  $\leftarrow$  Infer(node.qs, action);
24     Create new child node with state (qs', next_pose, qo', G) ;
25   return node ;
26 Function Simulation(node, lookahead_policy):
27   foreach action in node.possible_actions do
28     action  $\leftarrow$  choice from node.actions;
29     next_pose, qs', qo', G  $\leftarrow$  Infer(qs, action, lookahead_policy);
30     if best measured G then
31       best_G  $\leftarrow$  G;
32   total_reward  $\leftarrow$  total_reward + best_G;
33   return total_reward ;
34 Function Backpropagate(node, reward):
35   while node exists do
36     node  $\leftarrow$  update node(node, reward);
37     node  $\leftarrow$  node.parent;
38 Function SelectBestAction(root):
39   foreach (a, child) in root.childs do
40     Compute average reward of child;
41   return action action with highest AIF policy score ;
42 Function Infer(qs, action):
43   next_pose  $\leftarrow$  Transition(pose, action);
44   qs'  $\leftarrow$  BeliefTransition(qs, action);
45   qo'  $\leftarrow$  ExpectedObservation(qs');
46   G  $\leftarrow$  ExpectedFreeEnergy(qs', qo', qs, action);
47   return next_pose, qs', qo', G ;

```

Table D3 Exploration efficiency metrics across environments. CE: Coverage Efficiency (m^2/m), nAUC: Normalised Area Under Coverage Curve. Values are mean $\pm$ std over 5 runs.

Env.	Model	CE	nAUC
Small Warehouse	Manual	3.13 $\pm$ 0.43	0.74 $\pm$ 0.03
	AIMAPP	2.25 $\pm$ 0.87	0.87 $\pm$ 0.04
	Frontiers	1.08 $\pm$ 0.12	0.73 $\pm$ 0.03
	FAEL	2.60 $\pm$ 0.58	0.98 $\pm$ 0.03
	GBPlanner	1.59 $\pm$ 0.25	0.63 $\pm$ 0.06
Mini warehouse	Manual	2.66 $\pm$ 0.16	0.92 $\pm$ 0.01
	AIMAPP	2.64 $\pm$ 0.98	0.91 $\pm$ 0.04
	Frontiers	2.45 $\pm$ 0.07	0.88 $\pm$ 0.01
	FAEL	2.61 $\pm$ 1.16	0.85 $\pm$ 0.03
	GBPlanner	2.55 $\pm$ 0.4	0.82 $\pm$ 0.08
Real Garage	Manual	13.18	0.74
	AIMAPP	7.41 $\pm$ 0.87	0.71 $\pm$ 0.03

Table D4 Number of times the robot got stuck and required intervention per environment.

External intervention	AIMAPP	FAEL	Gbplanner	Frontiers
Home	2	5	1	5
Big warehouse	0	5	2	0
Small warehouse	1	3	4	2
Mini warehouse	0	2	2	3
Real home	4	x	x	x
Real parking	2	x	x	x
Real warehouse	7	x	x	14

References

- [1] Hewawasam, H.S., Ibrahim, M.Y., Appuhamillage, G.K.: Past, present and future of path-planning algorithms for mobile robot navigation in dynamic environments. *IEEE Open Journal of the Industrial Electronics Society* **3**, 353–365 (2022) <https://doi.org/10.1109/OJIES.2022.3179617>
- [2] Dehghani Tezerjani, M., Khoshnazar, M., Tangestanizade, M., Yang, Q.: A Survey on Reinforcement Learning Applications in SLAM (2024). <https://doi.org/10.48550/arXiv.2408.14518>
- [3] Tinguy, D., Remmery, S., Mazzaglia, P., Verbelen, T., Dhoedt, B.: Learning to Navigate from Scratch using World Models and Curiosity: the Good, the Bad, and the Ugly (2023). <https://arxiv.org/abs/2308.15852>
- [4] Campos, C., Elvira, R., Gomez, J.J., Montiel, J.M.M., Tardos, J.D.: ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM. *IEEE Transactions on Robotics* **37**(6), 1874–1890 (2021)
- [5] Xu, W., Cai, Y., He, D., Lin, J., Zhang, F.: FAST-LIO2: fast direct lidar-inertial odometry. *CoRR* **abs/2107.06829** (2021) [2107.06829](https://arxiv.org/abs/2107.06829)
- [6] An, D., Wang, H., Wang, W., Wang, Z., Huang, Y., He, K., Wang, L.: ETPNav: Evolving Topological Planning for Vision-Language Navigation in Continuous Environments (2024). <https://arxiv.org/abs/2304.03047>
- [7] Topiwala, A., Inani, P., Kathpal, A.: Frontier Based Exploration for Autonomous Robot (2018). <https://doi.org/10.13140/RG.2.2.34130.40641> . <https://arxiv.org/abs/1806.03581>
- [8] Chaplot, D.S., Gandhi, D., Gupta, S., Gupta, A., Salakhutdinov, R.: Learning to explore using active neural slam. In: *International Conference on Learning Representations (ICLR)* (2020)
- [9] Shah, D., Levine, S.: Viking: Vision-based kilometer-scale navigation with geographic hints. In: *Robotics: Science and Systems XVIII. Robotics: Science and Systems Foundation*, ??? (2022). <https://doi.org/10.15607/rss.2022.xviii.019> . <http://dx.doi.org/10.15607/RSS.2022.XVIII.019>
- [10] Sridhar, A., Shah, D., Glossop, C., Levine, S.: NoMaD: Goal Masked Diffusion Policies for Navigation and Exploration (2023). <https://arxiv.org/abs/2310.07896>
- [11] Parr, T., Pezzulo, G., Friston, K.: *Active Inference: The Free Energy Principle in Mind, Brain, and Behavior*. The MIT Press, ??? (2022). <https://doi.org/10.7551/mitpress/12441.001.0001>

- [12] Friston, K., Moran, R.J., Nagai, Y., Taniguchi, T., Gomi, H., Tenenbaum, J.: World model learning and inference. *Neural Networks* **144**, 573–590 (2021) <https://doi.org/10.1016/j.neunet.2021.09.011>
- [13] Tinguy, D., Verbelen, T., Dhoedt, B.: Learning dynamic cognitive map with autonomous navigation. *Frontiers in Computational Neuroscience* **18** (2024) <https://doi.org/10.3389/fncom.2024.1498160>
- [14] Huang, J., Zhou, B., Fan, Z., Zhu, Y., Jie, Y., Li, L., Cheng, H.: Fael: Fast autonomous exploration for large-scale environments with a mobile robot. *IEEE Robotics and Automation Letters* **8**, 1667–1674 (2023)
- [15] Dang, T., Tranzatto, M., Khattak, S., Mascarich, F., Alexis, K., Hutter, M.: Graph-based subterranean exploration path planning using aerial and legged robots. *Journal of Field Robotics* **37**(8), 1363–1388 (2020). Wiley Online Library
- [16] nav2: nav2. Accessed: 2024-12-01 (2021). <https://docs.nav2.org/>
- [17] Shu, F., Wang, J., Pagani, A., Stricker, D.: Structure PLP-SLAM: Efficient Sparse Mapping and Localization using Point, Line and Plane for Monocular, RGB-D and Stereo Cameras (2023). <https://arxiv.org/abs/2207.06058>
- [18] Wang, H., Wang, C., Xie, L.: Lightweight 3-d localization and mapping for solid-state lidar. *IEEE Robotics and Automation Letters* **6**(2), 1801–1807 (2021) <https://doi.org/10.1109/LRA.2021.3060392>
- [19] Qin, H., Shao, S., Wang, T., Yu, X., Jiang, Y., Cao, Z.: Review of autonomous path planning algorithms for mobile robots. *Drones* **7**(3) (2023) <https://doi.org/10.3390/drones7030211>
- [20] Jardali, H., Ali, M., Liu, L.: Autonomous mapless navigation on uneven terrains. 2024 IEEE International Conference on Robotics and Automation (ICRA), 13227–13233 (2024)
- [21] Sharma, S., Curtis, A., Kryven, M., Tenenbaum, J.B., Fiete, I.R.: Map induction: Compositional spatial submap learning for efficient exploration in novel environments. *CoRR* **abs/2110.12301** (2021) [2110.12301](https://arxiv.org/abs/2110.12301)
- [22] Milford, M.J., Wyeth, G.F., Prasser, D.: RatSLAM: a hippocampal model for simultaneous localization and mapping. In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 1, pp. 403–4081 (2004). <https://doi.org/10.1109/ROBOT.2004.1307183>
- [23] Guo, Z.D., Thakoor, S., Píslar, M., Pires, B.A., Althé, F., Tallec, C., Saade, A., Calandriello, D., Grill, J.-B., Tang, Y., Valko, M., Munos, R., Azar, M.G., Piot, B.: BYOL-Explore: Exploration by Bootstrapped Prediction (2022). <https://arxiv.org/abs/2206.08332>

- [24] Shah, D., Eysenbach, B., Kahn, G., Rhinehart, N., Levine, S.: Rapid Exploration for Open-World Navigation with Latent Goal Models (2023). <https://arxiv.org/abs/2104.05859>
- [25] Mendonca, R., Rybkin, O., Daniilidis, K., Hafner, D., Pathak, D.: Discovering and Achieving Goals via World Models (2021). <https://arxiv.org/abs/2110.09514>
- [26] Liu, Y., Ni, R., Qureshi, A.H.: Physics-informed neural mapping and motion planning in unknown environments. *IEEE Transactions on Robotics* **41**, 2200–2212 (2025) <https://doi.org/10.1109/tro.2025.3548495>
- [27] Li, J., Xu, Z., Zhu, D., Dong, K., Yan, T., Zeng, Z., Yang, S.X.: Bio-inspired intelligence with applications to robotics: a survey. *Intelligence and Robotics* **1**(1) (2021) <https://doi.org/10.20517/ir.2021.08>
- [28] Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A.J., Banino, A., Denil, M., Goroshin, R., Sifre, L., Kavukcuoglu, K., Kumaran, D., Hadsell, R.: Learning to navigate in complex environments. *CoRR* **abs/1611.03673** (2016) [1611.03673](https://arxiv.org/abs/1611.03673)
- [29] Lanillos, P., Meo, C., Pezzato, C., Meera, A.A., Baioumy, M., Ohata, W., Tschantz, A., Millidge, B., Wisse, M., Buckley, C.L., Tani, J.: Active Inference in Robotics and Artificial Agents: Survey and Challenges (2021). <https://arxiv.org/abs/2112.01871>
- [30] Kaplan, R., Friston, K.J.: Planning and navigation as active inference. *Biol. Cybern.* **112**(4), 323–343 (2018) <https://doi.org/10.1007/s00422-018-0753-2>
- [31] Zhao, M.: Human spatial representation: What we cannot learn from the studies of rodent navigation. *Journal of Neurophysiology* **120** (2018) <https://doi.org/10.1152/jn.00781.2017>
- [32] Neacsu, V., Mirza, M.B., Adams, R.A., Friston, K.J.: Structure learning enhances concept formation in synthetic active inference agents. *PLOS ONE* **17**(11), 1–34 (2022) <https://doi.org/10.1371/journal.pone.0277199>
- [33] Friston, K., Da Costa, L., Hafner, D., Hesp, C., Parr, T.: Sophisticated inference. *Neural Computation* **33**, 713–763 (2021) <https://doi.org/10.1162/neco.a.01351>
- [34] Fountas, Z., Sajid, N., Mediano, P.A.M., Friston, K.: Deep active inference agents using monte-carlo methods. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) *Advances in Neural Information Processing Systems*, vol. 33, pp. 11662–11675. Curran Associates, Inc., ??? (2020). <https://proceedings.neurips.cc/paper/2020/file/865dfbde8a344b44095495f3591f7407-Paper.pdf>
- [35] Safron, A., Çatal, O., Verbelen, T.: Generalized simultaneous localization and mapping (g-slam) as unification framework for natural and artificial intelligences:

towards reverse engineering the hippocampal/entorhinal system and principles of high-level cognition. *Frontiers in Systems Neuroscience* **Volume 16 - 2022** (2022) <https://doi.org/10.3389/fnsys.2022.787659> . Moran, Rosalyn J.

- [36] Nozari, S., Krayani, A., Marin, P., Marcenaro, L., Martín Gómez, D., Regazzoni, C.: Exploring action-oriented models via active inference for autonomous vehicles. *EURASIP Journal on Advances in Signal Processing* **2024** (2024) <https://doi.org/10.1186/s13634-024-01173-9>
- [37] de Tinguy, Daria and Van de Maele, Toon and Verbelen, Tim and Dhoedt, Bart: Spatial and temporal hierarchy for autonomous navigation using active inference in minigrid environment. *ENTROPY* **26**(1), 32 (2024)
- [38] Peer, M., Brunec, I.K., Newcombe, N.S., Epstein, R.A.: Structuring knowledge with cognitive maps and cognitive graphs. *Trends in Cognitive Sciences* **25**(1), 37–54 (2021) <https://doi.org/10.1016/j.tics.2020.10.004>
- [39] Foo, P., Warren, W., Duchon, A., Tarr, M.: Do humans integrate routes into a cognitive map? map- versus landmark-based navigation of novel shortcuts. *Journal of experimental psychology. Learning, memory, and cognition* **31**, 195–215 (2005) <https://doi.org/10.1037/0278-7393.31.2.195>
- [40] Epstein, R., Patai, E.Z., Julian, J., Spiers, H.: The cognitive map in humans: Spatial navigation and beyond. *Nature Neuroscience* **20**, 1504–1513 (2017) <https://doi.org/10.1038/nn.4656>
- [41] Warren, W.H., Rothman, D.B., Schnapp, B.H., Ericson, J.D.: Wormholes in virtual space: From cognitive maps to cognitive graphs. *Cognition* **166**, 152–163 (2017) <https://doi.org/10.1016/j.cognition.2017.05.020>
- [42] Bush, D., Barry, C., Manson, D., Burgess, N.: Using grid cells for navigation. *Neuron* **87**, 507–520 (2015)
- [43] Rosenberg, M., Zhang, T., Perona, P., Meister, M.: Mice in a labyrinth show rapid learning, sudden insight, and efficient exploration. *eLife* **10**, 66175 (2021) <https://doi.org/10.7554/eLife.66175>
- [44] Browne, C.B., Powley, E., Whitehouse, D., Lucas, S.M., Cowling, P.I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., Colton, S.: A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games* **4**(1), 1–43 (2012) <https://doi.org/10.1109/TCIAIG.2012.2186810>
- [45] Friston, K.J., Salvatori, T., Isomura, T., Tschantz, A., Kiefer, A., Verbelen, T., Koudahl, M., Paul, A., Parr, T., Razi, A., Kagan, B., Buckley, C.L., Ramstead, M.J.D.: Active Inference and Intentional Behaviour (2023). <https://arxiv.org/abs/2312.07547>

- [46] Friston, K., Parr, T., Zeidman, P.: Bayesian model reduction (2019) [arXiv:1805.07092](https://arxiv.org/abs/1805.07092) [stat.ME]
- [47] Friston, K.J., Costa, L.D., Tschantz, A., Kiefer, A., Salvatori, T., Neacsu, V., Koudahl, M., Heins, C., Sajid, N., Markovic, D., Parr, T., Verbelen, T., Buckley, C.L.: Supervised structure learning (2023) [arXiv:2311.10300](https://arxiv.org/abs/2311.10300) [cs.LG]
- [48] aws-robotics: aws-robomaker-small-warehouse-world. Accessed: 2024-08-01 (2020). <https://github.com/aws-robotics/aws-robomaker-small-warehouse-world>
- [49] aws-robotics: aws-robomaker-small-house-world. Accessed: 2024-10-01 (2021). <https://github.com/aws-robotics/aws-robomaker-small-house-world>
- [50] Gyanani, P., Agarwal, M., Osari, R., et al.: Autonomous mobile vehicle using ros2 and 2d-lidar and slam navigation. Research Square **Preprint (Version 1)** (2024) <https://doi.org/10.21203/rs.3.rs-4323431/v1> . Available at Research Square
- [51] Tranzatto, M., Dharmadhikari, M., Bernreiter, L., Camurri, M., Khattak, S., Mascari, F., Pfreundschuh, P., Wisth, D., Zimmermann, S., Kulkarni, M., Reijgwart, V., Casseau, B., Homberger, T., Petris, P.D., Ott, L., Tubby, W., Waibel, G., Nguyen, H., Cadena, C., Buchanan, R., Wellhausen, L., Khedekar, N., Andersson, O., Zhang, L., Miki, T., Dang, T., Mattamala, M., Montenegro, M., Meyer, K., Wu, X., Briod, A., Mueller, M., Fallon, M., Siegwart, R., Hutter, M., Alexis, K.: Team CERBERUS Wins the DARPA Subterranean Challenge: Technical Overview and Lessons Learned (2022). <https://arxiv.org/abs/2207.04914>
- [52] Oleynikova, H., Taylor, Z., Fehr, M., Nieto, J.I., Siegwart, R.: Voxblox: Building 3d signed distance fields for planning. CoRR **abs/1611.03631** (2016) [1611.03631](https://arxiv.org/abs/1611.03631)
- [53] Duberg, D., Jensfelt, P.: UFOMap: An efficient probabilistic 3D mapping framework that embraces the unknown. IEEE Robotics and Automation Letters **5**(4), 6411–6418 (2020) <https://doi.org/10.1109/LRA.2020.3013861>
- [54] Xu, H., Baracska, P., O’Neill, J., Csicsvari, J.: Assembly responses of hippocampal ca1 place cells predict learned behavior in goal-directed spatial tasks on the radial eight-arm maze. Neuron **101**(1), 119–1324 (2019) <https://doi.org/10.1016/j.neuron.2018.11.015>
- [55] Johnson, A., Redish, A.D.: Neural ensembles in ca3 transiently encode paths forward of the animal at a decision point. Journal of Neuroscience **27**(45), 12176–12189 (2007) <https://doi.org/10.1523/JNEUROSCI.3761-07.2007> <https://www.jneurosci.org/content/27/45/12176.full.pdf>
- [56] Kay, K., Chung, J.E., Sosa, M., Schor, J.S., Karlsson, M.P., Larkin, M.C., Liu, D.F., Frank, L.M.: Constant sub-second cycling between representations of possible futures in the hippocampus. Cell **180**(3), 552–56725 (2020) <https://doi.org/10.1016/j.cell.2020.03.045>

[//doi.org/10.1016/j.cell.2020.01.014](https://doi.org/10.1016/j.cell.2020.01.014)

- [57] Stachenfeld, K., Botvinick, M., Gershman, S.: The hippocampus as a predictive map. *Nature Neuroscience* **20**(11), 1643–1653 (2017) <https://doi.org/10.1101/097170>
- [58] Milford, M.: Vision-based place recognition: how low can you go? *The International Journal of Robotics Research* **32**(7), 766–789 (2013) <https://doi.org/10.1177/0278364913490323> <https://doi.org/10.1177/0278364913490323>
- [59] Chaplot, D.S., Gandhi, D., Gupta, A., Salakhutdinov, R.: Object Goal Navigation using Goal-Oriented Semantic Exploration (2020). <https://arxiv.org/abs/2007.00643>
- [60] Dorbala, V.S., Mullen, J.F., Manocha, D.: Can an embodied agent find your “cat-shaped mug”? llm-based zero-shot object navigation. *IEEE Robotics and Automation Letters* **9**(5), 4083–4090 (2024) <https://doi.org/10.1109/lra.2023.3346800>
- [61] Wakayama, S., Candela, A., Hayne, P., Ahmed, N.: Active inference for bandit-based autonomous robotic exploration with dynamic preferences. *IEEE Transactions on Robotics* **41**, 3841–3851 (2025) <https://doi.org/10.1109/TRO.2025.3577041>
- [62] Catal, O., Verbelen, T., Van de Maele, T., Dhoedt, B., Safron, A.: Robot navigation as hierarchical active inference. *Neural Networks* **142**, 192–204 (2021) <https://doi.org/10.1016/j.neunet.2021.05.010>
- [63] Caticha, A.: The information geometry of space and time. In: *AIP Conference Proceedings*, vol. 803, pp. 355–365. AIP, ??? (2005). <https://doi.org/10.1063/1.2149814> . <http://dx.doi.org/10.1063/1.2149814>
- [64] Hwang, J., Hong, Z.-W., Chen, E., Boopathy, A., Agrawal, P., Fiete, I.: Grid Cell-Inspired Fragmentation and Recall for Efficient Map Building (2024). <https://arxiv.org/abs/2307.05793>
- [65] Selin, M., Tiger, M., Duberg, D., Heintz, F., Jensfelt, P.: Efficient autonomous exploration planning of large-scale 3-d environments. *IEEE Robotics and Automation Letters* **4**(2), 1699–1706 (2019) <https://doi.org/10.1109/LRA.2019.2897343>
- [66] Nuijten, W.W.L., Lukashchuk, M., Laar, T., Vries, B.: A Message Passing Realization of Expected Free Energy Minimization (2025). <https://arxiv.org/abs/2508.02197>
- [67] Turtlebot: Turtlebot versions. Accessed: 2024-12-16 (2024). <https://www.turtlebot.com/about/>

- [68] clearpathrobotics: Jackal. Accessed: 2025-07-28. <https://clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/>
- [69] husarion: rosbotoXL. Accessed: 2025-07-28. <https://husarion.com/manuals/rostop-xl/>