Think Fast and Far: Long-Horizon Online POMDP Planning via Rapid State Sampling

Journal Title
XX(X):1–19

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Yuanchu Liang[1], Edward Kim[1], J. Arden Knoll[2], Wil Thomason[2], Zachary Kingston[2], Lydia E. Kavraki[2] and Hanna Kurniawati[1]

Abstract

Partially Observable Markov Decision Processes (POMDPs) are a general and principled framework for motion planning under uncertainty. Despite tremendous improvement in the scalability of POMDP solvers, long-horizon POMDPs remain difficult to solve. To alleviate the difficulty, this paper proposes a new approximate online POMDP solver, called Reference-Based Online POMDP Planning via Rapid State Space Sampling (ROP-RaS3). ROP-RaS3 uses novel extremely fast sampling-based motion planning techniques to sample the state space and generate a diverse set of macro actions online, which are then used to bias belief-space sampling and infer high-quality policies *without* requiring exhaustive enumeration of the action space—a fundamental constraint for modern online POMDP solvers. ROP-RaS3 converges to a near-optimal reference-based solution at a rate that depends on the number of sampled actions, rather than the size of the action space. ROP-RaS3 is evaluated on various long-horizon POMDPs with up to 3000 lookahead steps and 35-dimensional state spaces, where the state, action and observation spaces can be continuous, discrete, or a hybrid of discrete and continuous. Although the reference-based optimal solution may not be the same as the optimal POMDP solution, empirical results indicate that in all of these problems, in terms of success rate, ROP-RaS3 *outperforms* other state-of-the-art methods by up to *multiple folds*. We also demonstrate the capability of our approach on a physical robot demonstration. This work extends the theory and empirical results of our ISRR24 paper. Code can be found at `https://github.com/RDLLab/ROPRAS3`.

Keywords

Motion Planning, Sampling-Based Motion Planning, Planning under Uncertainty, POMDP, Hardware Acceleration

1 Introduction

Motion planning in partially observed and non-deterministic environments is a critical component of reliable and robust robot operation. Partially Observable Markov Decision Processes (POMDPs) (Smallwood and Sondik 1973; Kaebling et al. 1998) are a natural way to formulate such problems. The key insight of the POMDP framework is to represent uncertainty on the effects of actions, perceptions and initial states as probability distributions, and then reason about the best strategy to perform with respect to distributions over the problem's state space, called *beliefs*, rather than the state space itself.

Although POMDPs' methodical reasoning about uncertainty comes at the cost of high computational complexity (Papadimitriou and Tsitsiklis 1987), the POMDP framework is practical for many robotics problems, thanks in large part to sampling-based approaches. These approaches relax the problem of finding an optimal solution to an approximate one by sampling states from the belief space and computing the best action from only the samples. Scalable anytime methods under this approach (surveyed by Kurniawati (2022)) have been proposed for solving large POMDP problems. However, computing good solutions to long-horizon (*e.g.*, ≥ 15 look-ahead steps) POMDPs remains difficult. Here, look-ahead steps refer to the minimum number of actions needed for the agent to identify which action is most beneficial to reach the goal under the partially observable characteristics of the problem.

Early results from the literature (Kurniawati et al. 2011) indicate that Sampling-Based Motion Planning (SBMP)—sampling-based approaches designed for deterministic motion planning—can be used to tackle the challenges of long-horizon problems in offline POMDP planning. Specifically, SBMPs can be used to generate suitable macro-actions (*i.e.*, sequences of actions) to reduce the effective planning horizon for a POMDP solver. Macro-actions generated via SBMP automatically adapt to geometric features of the valid region and tend to cover diverse paths in the state space.

Although this approach performs well for offline POMDP planning, it is often impractical for online planning for two reasons. First is the speed of SBMPs, which historically required hundreds of milliseconds to tens of seconds to find a single motion plan. Second, most online POMDP planners (Silver and Veness 2010; Kurniawati and Yadav 2013; Ye et al. 2017) exhaustively enumerate each action at each sampled belief in computing the best action to perform. Such enumerations limit the number of actions sampled at run time, hence restricting online planners to quickly cover a good reachable belief space, from which

[1]Australian National University, Canberra ACT 2601, Australia. Corresponding Author: `{hanna.kurniawati}@anu.edu.au` [2]Rice University, Houston TX 77005, USA.

the optimal solution can be computed efficiently (Lee et al. 2007). However, the recently proposed Vector-Accelerated Motion Planning (VAMP) framework (Thomason et al. 2024) enables SBMPs to find solutions on microsecond timescales, multiple orders of magnitude faster than prior approaches. Concurrently, recently proposed *reference-based* POMDP *planners* (Kim et al. 2023) remove the requirement for exhaustive enumeration of the entire action space to compute approximately optimal POMDP solutions.

Leveraging the above two advances, we propose an online POMDP solver, called Reference-Based Online POMDP Planning via Rapid State Space Sampling (ROP-RAS3), which is a reference-based POMDP planner that employs a VAMP-enhanced macro-action sampler as its underlying reference policy. We modify the formulation from Kim et al. (2023) so that the reference is defined as a policy instead of a belief-to-belief transition, which makes it easier to generate suitable macro-actions and apply them to the reference-based POMDP solving approach. We show that the modified reference-based Bellman backup naturally allows processing continuous action spaces by replacing the optimization procedures in the POMDP Bellman backup with expectations. By adopting the analysis strategy used in Lim et al. (2020, 2023), we show a convergence rate that depends on $C_{\mathcal{A}}$, the number of actions a planner samples at each belief node instead of $|\mathcal{A}|$, the size of the action space reported in Lim et al. (2020, 2023). Although an optimal solution of a reference-based POMDP can be different from the original POMDP, the empirical section of our work demonstrates how this formulation enables us to solve challenging robotics tasks. We evaluate ROP-RAS3 on multiple long-horizon POMDPs, including 4 navigation tasks and 3 manipulation tasks, which require planning horizons of hundreds to thousands of steps. Comparisons with state-of-the-art online POMDP planners—including POMCP (Silver and Veness 2010), a POMCP modification that uses VAMP to generate macro-actions, and DESPOT (Ye et al. 2017) with learned macro-actions (Lee et al. 2021; Liang and Kurniawati 2023)—indicate that ROP-RAS3 substantially outperforms state-of-the-art methods in all evaluation scenarios. Learning-based methods like MAGIC (Lee et al. 2021) do not naturally extend to high-dimensional motion planning domains, but ROP-RAS3 is able to solve problems with dimensionality up to 35. ROP-RAS3 is also deployed to a Hello-Robot Stretch 3 mobile base manipulator. In our pedestrian crossing scenario, ROP-RAS3 is the only method that demonstrates smart maneuvers to dodge a moving pedestrian and navigate to the goal. Our code will be released after publication.

Remark. This work extends our ISRR24 paper Liang et al. (2024). We modified the original algorithm to deal with continuous action and observation spaces and provided a neater way to do backups in the belief tree. The convergence analysis of ROP-RAS3 is added together with 3 new manipulation simulations and 1 physical robot demonstrations.

2 Background and Related Work

2.1 Sampling-Based Motion Planners

Sampling-based motion planning (SBMP) is a common, effective family of algorithms (*e.g.*, Kavraki et al. (1996); Kuffner and LaValle (2000)) for solving *deterministic* motion planning problems (LaValle 2006). They are able to find collision-free motions for high degree-of-freedom (DoF) robots in environments containing many obstacles by drawing samples from a robot's *configuration space*, the set of all possible robot configurations, $q \in Q$. Collisions between the robot and the environment or with itself partition the configuration space into *valid* (Q_{free}) and *invalid* ($Q \setminus Q_{\text{free}}$) configurations. A deterministic *motion planning problem* is then a tuple $(Q_{\text{free}}, q_I, Q_G)$ representing the task of finding a continuous path, $p : [0,1] \to Q_{\text{free}}$, from an initial configuration, q_I, to a goal region, $Q_G \subseteq Q_{\text{free}}$ (*i.e.*, $p(0) = q_I$ and $p(1) \in Q_G$). SBMPs solve such problems by building a discrete approximation of Q_{free} as a graph or tree connecting sampled configurations in Q_{free} by short, local motions. Once both q_I and Q_G are connected by this structure, a valid robot motion plan can be found with graph search methods. The solutions are deterministic, open-loop plans, and may not be robust against uncertainties or changes in configuration spaces.

2.2 POMDP Background

An infinite-horizon POMDP is defined as the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \mathcal{Z}, R, \gamma, b_0 \rangle$ where $\mathcal{S}$ denotes the set of all possible states of the system being considered, $\mathcal{A}$ denotes the set of all possible actions, and $\mathcal{O}$ denotes the set of all possible observations. We assume measures are properly defined for these spaces. In our context, the state space encompasses the robot's configuration space ($\mathcal{S} \equiv Q$). The transition function $\mathcal{T}(s' \mid s, a)$ is the conditional probability that the robot will be in state $s' \in \mathcal{S}$ after performing action $a \in \mathcal{A}$ from state $s \in \mathcal{S}$. The observation function $\mathcal{Z}(o \mid s', a)$ is the conditional probability that the agent perceives $o \in \mathcal{O}$ when it is in state $s' \in \mathcal{S}$ after performing action $a \in \mathcal{A}$. The reward is a bounded real-valued function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. The parameter $\gamma \in (0,1)$ is the discount factor which ensures the objective function is well-defined.

In general, the robot does not know the true state. At each time-step, the robot maintains a *belief* about its state, which is a probability distribution over the state space. The space of all possible beliefs is denoted by $\mathcal{B} := \Delta(\mathcal{S})$. The agent starts from a given initial belief, denoted as b_0.

If b' denotes the agent's next belief after taking an action $a \in \mathcal{A}$ and receiving the corresponding observation $o \in \mathcal{O}$, then the updated belief is given by

$$b'(s') = \tau(b, a, o) \propto \mathcal{Z}(o \mid s', a) \int_{\mathcal{S}} \mathcal{T}(s' \mid s, a)\, b(s) \mathrm{d}s$$

For any given belief b and action a, the expected reward is given by $R(b,a) := \int_{s \in \mathcal{S}} R(s,a) b(s) \mathrm{d}s$. A (stochastic) *policy* is a mapping $\pi : \mathcal{B} \to \Delta(\mathcal{A})$. We denote its distribution for any given input $b \in \mathcal{B}$ by $\pi(\cdot \mid b)$. A policy is *deterministic* if it only has support at a single point $a \in \mathcal{A}$. Let Π be the class of all policies.

Given a policy $\pi \in \Pi$, we define the *value function* $V^\pi : \mathcal{B} \to \mathbb{R}$ to be the expected total discounted reward,

$V^{\pi}(b) := \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(b_t, a_t))]$, where the expectation is taken with respect to the policy and the transition probability.

In this paper, a *solution* to the POMDP is a deterministic policy $\pi^* \in \Pi$ satisfying

$$V^{\pi^*}(b) = \sup_{\pi \in \Pi} V^{\pi}(b)$$

on $\mathcal{B}$. The Bellman equation

$$V(b) = \max_{a \in \mathcal{A}} \left[R(b,a) + \gamma \int_{\mathcal{O}} P(o \mid a, b) V\big(\tau(b,a,o)\big) \mathrm{d}o \right] \tag{1}$$

is satisfied by the optimal value function V^{π^*}. The intrinsic conditional probability

$$P(o \mid a, b) := \int_{\mathcal{S}} \mathcal{Z}(o \mid s', a) \left(\int_{\mathcal{S}} \mathcal{T}(s' \mid s, a) b(s) \mathrm{d}s \right) \mathrm{d}s'$$

is the probability that the agent perceives o, having performed the action a, under the belief b.

2.3 Long-Horizon POMDPs

Solving long-horizon POMDPs is a significant challenge: the branching factor as a result of actions and observation grows exponentially with respect to the planning horizon. Planning over *macro-actions* (i.e., a set of action sequences) is a common approach to deal with long-horizon POMDPs (Theocharous and Kaelbling 2003; He et al. 2010; Kurniawati et al. 2011; Flaspohler et al. 2020; Lee et al. 2021; Liang and Kurniawati 2023). Although this reduces the effective planning horizon, choosing a set of good macro-actions for planning is critical and automatic construction of macro-actions can require significant effort; *e.g.*, the learning time for MAGIC (Lee et al. 2021) can be on the order of hours.

Irrespective of macro-action use, most online POMDP planners (Silver and Veness 2010; Kurniawati and Yadav 2013; Ye et al. 2017) exhaustively enumerate all actions from each sampled belief in order to estimate the optimal action. Many efficient optimization methods, which are generally based on gradient ascent, cannot be used effectively because computing the gradient of the POMDP value function is very expensive. Therefore, existing planners seldom explore long-term information and tend to perform poorly for horizons greater than 15 steps.

Recently, Kim et al. (2023) have mitigated this limitation by solving a *reference-based* POMDP: a reformulated POMDP whose reward objective incurs a KL-penalty for deviating too far from a given stochastic *reference belief-to-belief transition*: $\bar{U}(a, o \mid b) := P(o \mid a, b)\bar{\pi}(a \mid b)$ where $\bar{\pi}$ is a given stochastic reference policy. The reward with respect to U is defined as $R(b, U) := \int_{\mathcal{A},\mathcal{O}} R(b,a)U(a, o \mid b)\mathrm{d}a\mathrm{d}o$. The value of a belief is given by

$$\mathcal{V}^*(b) = \sup_{U \in \mathcal{U}(b)} \Big(R(b,U) - \mathrm{KL}(U \,\|\, \bar{U}) + \gamma \int_{\mathcal{A},\mathcal{O}} U(a, o \mid b) \mathcal{V}^*(\tau(b,a,o)) \mathrm{d}a\mathrm{d}o \Big) \tag{2}$$

where $\mathcal{U}(b) \subseteq \Delta(\mathcal{A} \times \mathcal{O})$ is the set of admissible transitions at belief b. The RHS can be optimized analytically, effectively removing the need for enumerative optimization in online POMDP solving and thereby reducing the branching factor of the belief tree. Preliminary results from Kim et al. (2023) indicate that policies generated by this procedure can outperform state-of-the-art POMDP benchmarks on long-horizon POMDPs. However, the assumption of having access to a belief-to-belief transition is very strong and cumbersome to work with in practice. Kim et al. (2023) also employs relatively crude reference policies and does not incorporate macro-actions into planning.

2.4 Convergence of Online Sampling-based POMDP Planners

Despite many empirical advances for online POMDP planning, little theoretical justification for the convergence rates of these planners has been provided. Recently, Lim et al. (2020, 2023) formally showed that a planner that uses observation likelihood weightings (a technique adopted by many online POMDP planners) can estimate Q-values accurately with a convergence rate $\mathcal{O}(|\mathcal{A}|(|\mathcal{A}|C_{\mathcal{S}})^D \exp(-t_{\max}C_{\mathcal{S}}))$, where $|\mathcal{A}|$ is the size of the action space, $C_{\mathcal{S}}$ is the number of state samples in a belief node, D is the depth of the belief tree and $t_{\max}$ is a problem-specific bounding constant. The key of their analysis is to treat observation likelihood weighting as a self-normalized (SN) importance sampling process (Shachter and Peot 1990). The latter aims to estimate the expectation of an arbitrary function $f(x)$ where x is drawn from distribution $\mathcal{P}$, while the estimator only has access to another distribution $\mathcal{Q}$ along with the importance weights $\omega_{\mathcal{P}/\mathcal{Q}} \propto \mathcal{P}/\mathcal{Q}$. The following three quantities related to importance sampling are frequently used,

$$\tilde{\omega}_{\mathcal{P}/\mathcal{Q}}(x) := \frac{\omega_{\mathcal{P}/\mathcal{Q}}(x)}{\sum_{i=1}^{N} \omega_{\mathcal{P}/\mathcal{Q}}(x_i)} \tag{3}$$

$$d_{\alpha}(\mathcal{P} \| \mathcal{Q}) := \mathbb{E}_{x \sim \mathcal{Q}}[\omega_{\mathcal{P}/\mathcal{Q}}(x)^{\alpha}] \tag{4}$$

$$\tilde{\mu}_{\mathcal{P}/\mathcal{Q}} := \sum_{i=1}^{N} \tilde{\omega}_{\mathcal{P}/\mathcal{Q}}(x_i) f(x_i) \tag{5}$$

Here, (3) is the SN importance weight, (4) is the Rényi divergence and (5) is the SN estimator. Central to their analysis is the following self-normalized d_{∞}-concentration bound, where d_{∞} is the infinite Rényi divergence.

Theorem 1. *[SN d_{∞}-Concentration Bound (Lim et al. 2020)]. Let $\mathcal{P}$ and $\mathcal{Q}$ be two probability measures on the measurable space $(\mathcal{X}, \mathcal{F})$ with $\mathcal{P} \ll \mathcal{Q}$ and $d_{\infty}(\mathcal{P}||\mathcal{Q}) < +\infty$. Let $x_1, \ldots, x_N$ be i.i.d. random variables sampled from $\mathcal{Q}$ and $f : \mathcal{X} \to \mathbb{R}$ be a bounded Borel function ($||f||_{\infty} < +\infty$). Then for any $\lambda > 0$ and N large enough such that $\lambda > ||f||_{\infty} d_{\infty}(\mathcal{P}||\mathcal{Q})/\sqrt{N}$, the following bound holds with probability at least $1 - 3\exp(-N \cdot t^2(\lambda, N))$:*

$$|\mathbb{E}_{x \sim \mathcal{P}}[f(x)] - \tilde{\mu}_{\mathcal{P}/\mathcal{Q}}| \leq \lambda$$

where $t(\lambda, N)$ is defined as,

$$t(\lambda, N) \equiv \frac{\lambda}{||f||_{\infty} d_{\infty}(\mathcal{P}||\mathcal{Q})} - \frac{1}{\sqrt{N}}$$

However, their analysis does not cover the case of continuous action spaces as exhaustively enumerating all

actions for maximization is impossible. A subsequent work (Lim et al. 2021) proposed a Voronoi progressive widening technique to extend the convergence analysis to continuous action spaces. Instead of relying on Voronoi optimistic optimization and its convergence, we show that the reference policy acts as a natural action sampler that removes the need of online Q-value optimization, hence replacing the $|\mathcal{A}|$ in the convergence rate with $C_{\mathcal{A}}$, the number of actions we sample.

3 Continuous Reference-Based POMDPs over Stochastic Actions

The concept of a reference-based POMDP was introduced in Kim et al. (2023) as a generalization to POMDPs of the MDP formulations using KL-penalization in Azar et al. (2012); Todorov (2006). One limitation is that the formulation given by (2) is somewhat artificial in that reference policies are stated with respect to *belief-to-belief* transitions (see discussion in §8 of Kim et al. (2023)). In this paper, we will use a more natural formulation of a reference-based POMDP over *stochastic policies* rather than *belief-to-belief transitions* as in Kim et al. (2023). This allows us to directly work with reference policies which are easy to define and handcraft compared to belief-to-belief transitions. Further, the latter is less realistic as one does not have the choice of picking specific observations that resulted in the desired transitions, but is allowed to choose heuristic reference policies. Note that in doing so, all the benefits of the formulation in Kim et al. (2023) are still preserved.

Specifically, a reference-based POMDP over stochastic actions is specified by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{Z}, \mathcal{T}, R, \gamma, \eta, \bar{\pi} \rangle$. The presentation here treats the state, action and observation spaces as continuous spaces and assumes that the associated Borel σ-algebra exists and probability measures are properly defined with respect to the underlying σ-algebra. In addition to the standard parameters, we have a *temperature* parameter $\eta > 0$ and a given (stochastic) *reference policy* $\bar{\pi}(\cdot \,|\, b)$. We assume that $\bar{\pi}(\cdot \mid b) > 0, \forall b \in \mathcal{B}$. The value $\mathcal{V}$ (distinguished from the POMDP V-value) of a reference-based POMDP for a given $b \in \mathcal{B}$ satisfies the reference-based Bellman equation,

$$\mathcal{V}(b) = \sup_{\pi \in \Pi} \Big[R(b,\pi) - \frac{1}{\eta} \mathrm{KL}(\pi \,\|\, \bar{\pi}) + \gamma \int_{\mathcal{A},\mathcal{O}} P(o \,|\, a, b) \pi(a \,|\, b) \mathcal{V}\big(\tau(b,a,o)\big) \,\mathrm{d}a \,\mathrm{d}o \Big] \quad (6)$$

where $R(b,\pi) := \int_{\mathcal{A},\mathcal{S}} R(s,a)\pi(a \,|\, b) b(s) \,\mathrm{d}a \,\mathrm{d}s$ is the reward estimate. A *solution* is a stochastic policy $\pi \in \Pi$ that maximizes $\mathcal{V}$. The problem can therefore be viewed as a KL-penalized POMDP whose objective is modified to trade off two (potentially competing) objectives: (1) abide by the reference policy, and (2) maximize reward. The trade-off is balanced by η and the *quality* of the reference policy—higher-quality reference policies are those that reduce the KL-divergence between the solution of the unpenalized POMDP (1) and the reference policy. The KL-penalty implies that $\pi(\cdot \mid b) \ll \bar{\pi}(\cdot \mid b)$ for any $b \in \mathcal{B}$. When the optimal policy is used as the reference policy, we trivially obtain the optimal policy as the solution, since the optimal policy is deterministic and the only solution that maximizes rewards and minimizes KL divergence is itself.

For an arbitrary $\bar{\pi}$, the supremum in (6) can be attained analytically by extending an argument of Azar et al. (2011, 2012) to POMDPs,

Theorem 2. *[Analytical Solution of Reference-based* POMDP*]. The exact solution of* (6) *is given by,*

$$\mathcal{V}(b) = \frac{1}{\eta} \log \Big[\int_{\mathcal{A}} \bar{\pi}(a \,|\, b) \exp\Big\{ \eta \mathcal{Q}(b,a) \Big\} \,\mathrm{d}a \Big]. \quad (7)$$

Moreover, the exact *solution of the Reference-based* POMDP *is given by,*

$$\pi^*(a \,|\, b) \propto \bar{\pi}(a \mid b) \exp(\eta \mathcal{Q}(b,a)) \quad (8)$$

In above theorem, we have defined the reference-based Q-value,

$$\mathcal{Q}(b,a) = R(b,a) + \int_{\mathcal{O}} P(o \mid a, b) \mathcal{V}(\tau(b,a,o)) \,\mathrm{d}o \quad (9)$$

Derivation details can be found in the Appendix A.1.1. The iterative backup procedure is guaranteed to converge to a unique solution $\mathcal{V}^*$ from which the policy can be read off using (8) Kim et al. (2023). The main point is that enumerative maximization in (1) is replaced with expectation.

This new backup brings multiple advantages. First, gradients of the Bellman equation in POMDPs are generally hard to compute because it requires an estimate of the expected total future reward. This difficulty in computing gradients makes numerical optimization in POMDP solving to generally be very computationally expensive. Our method removes this barrier and opens new ways to solve POMDPs. Specifically, it partially solves the optimization analytically, leaving numerical computation to only estimation of expectation, which can often be computed quickly via the Monte Carlo approach. Of course, if the probability distribution function is one that the Monte Carlo approach struggles with, such as those with rapidly oscillating functions or has very high variance, for instance, ROP-RAS3 requires much more careful consideration in its action selection (*i.e.*,, sampling strategy and selection of the reference policy). Second, the effective action space is reduced to the support of the reference policy, which can be much smaller than the entire action space, making ROP-RAS3 scalable to high-dimensional continuous action space.

These ideas are elaborated upon in the later sections, where we show that replacing exhaustive enumeration with Monte Carlo estimation provides a convergence rate that depends on $C_{\mathcal{A}}$, the number of actions sampled, rather than $|A|$, the size of the action space. Meanwhile, our algorithm ROP-RAS3, using the new backup, outperforms many baselines across diverse long horizon tasks.

Remark The optimal solution of a reference-based POMDP considered here is not the same as the optimal solution to the standard POMDP, as the objective functions considered are different. It is natural to ask whether it is possible to improve the reference policy over time by treating the newly obtained solution as the next step reference policy, leading to convergence to the optimal solution of the standard POMDP. The answer is positive, as presented in Kim and Kurniawati (2025). However, such a convergence requires

higher computational resources and is beyond the scope of this paper. In this work, we demonstrate how reference policies constructed based on fast motion plans in the state space can lead to scalable robust solutions for robots operating in a non-deterministic and partially observable world.

4 Algorithm

We introduce Reference-Based Online POMDP Planning via Rapid State Space Sampling (ROP-RAS3), our online anytime planner that uses sampling techniques to rapidly estimate $\mathcal{V}$-Values and $\mathcal{Q}$-Values for inferring good actions to execute. We motivate its design by briefly outlining VAMP's capability to induce high-quality reference policies. A general algorithm is proposed to utilize VAMP to infer actions for online POMDP planning. Then we elaborate ROP-RAS3 and analyze its online convergence rate.

4.1 Vector Accelerated Motion Planning (VAMP)

Towards fast deterministic motion planning, recent works have introduced new perspectives on *hardware-accelerated* sampling-based motion planning (SBMPs), using either CPU single-instruction, multiple-data (SIMD) (Thomason et al. 2024) or GPU single-instruction, multiple-thread (SIMT) (Sundaralingam et al. 2023) parallelism to find complete motion plans in tens of microseconds to tens of milliseconds. In particular, the authors of VAMP (Thomason et al. 2024) proposed a SIMD-vectorized approach to computing SBMP primitives (*i.e.*, local motion validation) that applies to all SBMPs thus, it is now possible to generate probabilistically-complete, global, collision-free trajectories for high-DoF systems at kilohertz rates—on the scale of tens of thousands of plans per second. The key insight of this work is to lift the "primitive" operations of the SBMP (e.g. forward kinematics and collision checking) to operate over *vectors* of configurations in parallel. Robot-specific code that uses these vector primitives is generated from a URDF using a tracing compiler—this pre-processing step is general to any robot. Functionally, this enables checking validity of a spatially distributed set of configurations over a candidate motion in parallel for the cost of a single collision check, massively lowering the expected time it takes to find colliding configurations along said motion. This development has called into question previously held perceptions that SBMPs are relatively time-expensive subroutines and—in the context of planning under uncertainty—opens the door to using SBMPs to guide POMDP planning on the fly.

4.2 POMDP Planning with SBMP-Generated Trajectories

A stochastic reference policy can be constructed from a deterministic policy. In robotics, such a representation can be naturally obtained by integrating deterministic motion plans with belief space sampling. We begin by presenting a general tree node expansion mechanism in Algorithm 1 to integrate belief-space sampling of a canonical online POMDP planner with state space sampling to generate macro-actions using SBMP. Since an expansion mechanism to get new actions for a new belief node is universal in online POMDP planning, Algorithm 1 can use any SBMP planner and can generalize to most existing online particle-based POMDP planners, such as POMCP (Silver and Veness 2010), DESPOT (Ye et al. 2017) and their derivatives (Kurniawati and Yadav 2013; Cai et al. 2021; Lee et al. 2021; Liang and Kurniawati 2023).

At a newly encountered belief node $\bar{b}$, SBMPEXPAND in Algorithm 1 aims to expand $C_{\mathcal{A}}$ of actions by firstly sampling a source state from the belief particles. In line 4, a target state is drawn from a distribution $\mathcal{J}(\cdot \mid \bar{b})$, a distribution over the state space $\mathcal{S}$ conditioned on the current belief particles. In robotics, target states can be states that reveal a certain amount of information to the robot, including states with high reward or penalty and states with observations. A deterministic motion plan is then constructed using any existing SBMP in line 6. The path is converted to a macro-action via PATHTOMACROACTION and added to the return list.

The choice of SBMP is an essential component for achieving high-quality policies online. VAMP enables rapid generation of collision-free state space paths to goals or to highly informative states, which in turn can be used as macro-actions for the POMDP planner. Thus, promising macro-actions can be dynamically created as a subroutine (line 5 in SBMPEXPANDTREE) *within* a POMDP planner itself in fractions of a second. In contrast, the state-of-the-art planner MAGIC (Lee et al. 2021) takes on the order of hours to learn macro-actions.

However, fast macro-action sampling alone is not sufficient, as current online POMDP planners perform numerical optimization via enumeration of *all* (macro) actions at each sampled belief, which results in a time and space complexity of $\mathcal{O}(|\vec{\mathcal{A}}|^h)$, where $\vec{\mathcal{A}}$ denotes the space of the macro-actions, h is the planning horizon. This complexity significantly limits the number of macro-actions and the depth of the tree it can construct, thereby significantly limiting the benefit of SBMP for POMDP solving. To tackle this problem, we propose ROP-RAS3 next that draws on insights from Section 4 while, in tandem, exploiting the speed of VAMP (Thomason et al. 2024)—an implementation of SIMD-vectorized SBMP—to create a rich set of diverse macro-actions, which the planner uses to efficiently explore relevant parts of the belief space.

4.3 ROP-RAS3

We propose ROP-RAS3 (Algorithm 2), a practical and scalable reference-based POMDP online planner for motion planning under uncertainty. We use arrows on top of letters to indicate macro elements associated to that letter (e.g. $\vec{a}$ is a macro action). With a slight abuse of notation, $\bar{b}$ denotes the belief node in a tree, $\bar{b}\vec{a}$ denotes the action node $\vec{a}$ associated with $\bar{b}$ and $\bar{b}\vec{a}\vec{o}$ is the next belief node given $\vec{a}$ and $\vec{o}$. The curly $\mathcal{C}(\cdot)$ indicates the children of a belief node (e.g. $\mathcal{C}(\bar{b})$ returns all actions associated with $\bar{b}$). And $N(\cdot)$ denotes the visitation count of a node. Values are initialized to zero by default.

4.3.1 Numerical Backup. Recall from Theorem 2, the reference-based Bellman backup can be broken down into two parts, the $\mathcal{Q}$-value defined in (9) and the $\mathcal{V}$-value defined

Algorithm 1 Online POMDP Planner Tree Expansions with SBMP-Generated Macro Actions

SBMPEXPAND($\bar{b}, C_{\mathcal{A}}$)

Initialize $l = \{\}$
for $i = 0, \ldots, C_{\mathcal{A}} - 1$ **do**
  sample source state, $s \sim \bar{b}$
  sample target state, $s' \sim \mathcal{J}(\cdot \mid \bar{b})$
  construct a motion plan, $p \leftarrow$ SBMP(s, s')
  convert a motion plan to a macro-action, $\vec{a} =$ PATHTOMACROACTION(p)
  append the new action, $l = l \cup \{\vec{a}\}$
end for
Return l

Algorithm 2 ROP-RAS3

Initialize tree T rooted at $\bar{b}$
while time permitting **do**
  SIMULATE($\bar{b}$)
end while
return T

SIMULATE($\bar{b}$)

Sample s from $\bar{b}$
if depth$(h) > D$ **then**
  return ROLLOUT$(\bar{b}, s)$
end if
$\vec{a} \leftarrow$ ACTIONPROGRESSIVEWIDEN$(\bar{b}, s)$
Sample $(s', \vec{o}, r(s, \vec{a}; \gamma))$ from generative model $\mathscr{G}(s, \vec{a})$
if $|\mathcal{C}(\bar{b}\vec{a}\vec{o})| \leq \beta_o N(\bar{b}\vec{a})^{\alpha_o}$ **then**
  $M(\bar{b}\vec{a}\vec{o}) \leftarrow M(\bar{b}\vec{a}\vec{o}) + 1$
else
  sample $\vec{o}$ from $\mathcal{C}(\bar{b}a)$ w.p $\frac{M(\bar{b}\vec{a}\vec{o})}{\sum_{\vec{o}} M(\bar{b}\vec{a}\vec{o})}$
end if
Create nodes for $\bar{b}\vec{a}$ and $\bar{b}\vec{a}\vec{o}$ if not created already
Add s' to belief particles of $\bar{b}\vec{a}\vec{o}$
return $\mathcal{V}(\bar{b}) \leftarrow$ BACKUP$(\bar{b}, \vec{a}, \vec{o}, r)$

ACTIONPROGRESSIVEWIDEN($\bar{b}, s$)

if $|\mathcal{C}(\bar{b})| \leq \beta_a(N(\bar{b})^{\alpha_a})$ **then**
  return SAMPLEMACROACTIONSBMP$(\bar{b}, s)$
end if
Sample $\vec{a}$ from $\bar{b}$.children uniformly
return $\vec{a}$

SAMPLEMACROACTIONSBMP($\bar{b}, s$)

Get the current configuration q_{start} from s (assert free)
$q_{\text{goal}} \leftarrow$ SAMPLEHEURISTICS$(\bar{b})$ (assert free)
Plan path p from q_{start} to q_{goal} using fast SBMP
return $\vec{a} =$ PATHTOMACROACTION(p)

BACKUP($\bar{b}, \vec{a}, \vec{o}, r$)

$r \rightarrow r + \gamma$SIMULATE$(b\vec{a}\vec{o})$
$N(\bar{b}) \leftarrow N(\bar{b}) + 1$
$N(\bar{b}\vec{a}) \leftarrow N(\bar{b}\vec{a}) + 1$
$\mathcal{Q}(\bar{b}\vec{a}) \leftarrow \mathcal{Q}(\bar{b}\vec{a}) + \frac{r - \mathcal{Q}(\bar{b}\vec{a})}{N(\bar{b}\vec{a})}$
$\mathcal{V}(\bar{b}) \leftarrow \frac{1}{\eta} \log \left[\exp(\eta \mathcal{V}(\bar{b})) + \frac{\exp(\eta \mathcal{Q}(\bar{b}\vec{a})) - \exp(\eta \mathcal{V}(\bar{b}))}{N(\bar{b})} \right]$
return $\mathcal{V}(\bar{b})$

in (7). The $\mathcal{Q}$-value can be viewed as an expected total future reward, $\mathcal{Q}(\bar{b}\vec{a}) \approx \frac{1}{N(\bar{b}\vec{a})} \sum_{i=1}^{N} R_i$, where R_i is the reward return from the i-th simulation at the node $\bar{b}\vec{a}$. From (7), by realizing that $\exp(\eta \mathcal{V}(b)) = \mathbb{E}[\exp(\eta \mathcal{Q}(b, a))]$, we can then estimate this term with $\frac{1}{N(\bar{b}\vec{a})} \sum_{i=1}^{N} \exp(\eta \mathcal{Q}(b, a))$. The iterative versions of both estimators are given in line 4 and 5 of the BACKUP function in Algorithm 2. Further details and convergence rates of such nested Monte Carlo Estimators can be found in Rainforth et al. (2018).

Due to the closed-form solution found in Theorem 2, we do not need to solve the bandit problem, which is typical of many online POMDP planners, dropping the need for UCB-like techniques. Second, without the need to expand all actions encountered at a given new node and restarting search from the root node, we can search deeper earlier by continuing to simulate from the newly sampled action, prioritizing long-horizon search commonly found in many robotics problems. Such tree search strategy is tightly linked to the backup equation (7).

4.3.2 Planning Tree Construction. ROP-RAS3 uses unweighted belief particles. This has the speed advantage of only retaining particles that have been sampled and obtained from the generative model during simulations. In SIMULATION, ROP-RAS3 aims to get an action and query the generative model to simulate for the next state, rewards and observations. Since both the action space and the observation space can be continuous, we apply the double progressive widening technique (Sunberg and Kochenderfer 2018) to ensure sampled elements can be revisited again in a new simulation. Our progressive widening technique is different to the standard implementation in the sense that we only need to uniformly sample from the existing actions and observations if the widenning condition is not met, thanks to Monte Carlo estimations in our backup procedures. If the condition of expanding a new action is met, then ROP-RAS3 proceeds to query a macro action $\vec{a}$ from a reference policy induced from a SBMP (see SAMPLEMACROACTIONSBMP), instead of uniformly sampling from the free action space as done in (Sunberg and Kochenderfer 2018). The generative model $\mathscr{G}$ is used to simulate for the next state, reward and observations. The macro action, observation, and the next state is added to the tree if they are not created yet. This repeats up to a predefined depth D. When the required depth is reached, ROP-RAS3 obtains an estimate of the

node's value by rollouts, a standard operation used by other online planners. The planner then approximates the exact backup (BACKUP) by carefully *maintaining* an empirical expectation, repeating backups on the simulated belief-tree path up to the root node. The above procedure is repeated until exhausting a computation budget (e.g., time), at which point the estimate of the optimal policy is read off from the tree's root node via (8).

4.3.3 Reference Policy. The simplicity offered by the formulation comes at a cost: if the optimal policy of the POMDP with an unmodified objective is too far from the reference policy (in the sense of the KL-divergence), pure reward maximization can be compromised. Of course, the optimal policy and hence this divergence are not known a priori. Instead ROP-RAS3 assumes that reference policies generated by accelerated SBMPs (VAMP) provide a reasonable starting point, leveraging them to rapidly sample high-quality deterministic policies to induce a reasonable partially observed reference policy which it deforms online. The subroutine SAMPLEMACROACTIONSBMP acts as the foundation for our reference policies. It lifts VAMP to belief space by sampling a state from the input belief particles, pick a target state with information (observation or reward) and query VAMP to build a deterministic motion plan connecting the two states. This motion plan is then converted to macro actions for ROP-RAS3. The target states can be either uniformly drawn from the free space or dynamically sampled based on the belief of the agent. Sampling these informative states ensure our planning tree only cover a compact reachable belief space, retaining optimal solution while reducing the search complexity as uninformative path occurs less frequently.

We emphasize that this modification is not negligible. Indeed, our results show that the performance deteriorates with problem complexity if the agent only executes the SBMP reference policy as an open-controller without planning for uncertainty. More implementation details are discussed in Section 5 and our code website.

4.4 Convergence Analysis

With the maximization steps in POMDP planning being replaced by an expectation estimation, ROP-RAS3's convergence can be analyzed by bounding $\mathcal{Q}$-value estimations and $\mathcal{V}$-Value estimations at all nodes of the tree. To make analysis tractable, we simplify ROP-RAS3 and distill its core computations into a theoretical algorithm, Reference-based Sparse Sampling (Algorithm 3). At the core of RBSS is a recursive self-normalized importance sampling process to estimate the beliefs based on the observation weightings. At a belief node, RBSS expands all sampled actions and observations to obtain a full belief particle set in the next step. This does not have the same computational advantage of only updating those particles that are sampled for the next step simulation as done in ROP-RAS3, but allows us to leverage Theorem 1 to provide anytime estimation error bounds at all nodes. Further, by noticing that the reference policy in ROP-RAS3 is created from deterministic motion plans by sampling the source state from the belief and the target state from a heuristic, we assume that the reference policy $\bar{\pi}$ used in RBSS is also created from an underlying fully observable policy $\pi^{\text{fo}} : \mathcal{S} \to \Delta\mathcal{A}$ as a pushforward measure of $b(s)$,

$$\bar{\pi}(\cdot \mid b) = \int_{\mathcal{S}} \delta_{\pi^{\text{fo}}(s)}(\cdot) b(s)\, \mathrm{d}s \tag{10}$$

where δ denotes the Dirac measure. Such representation allows us to compute the weights of an action based on the particle representation of the belief. More details about RBSS are discussed in the Appendix A.1.2. Theorem 3 shows RBSS has a convergence rate of $\mathcal{O}(C_{\mathcal{A}}(C_{\mathcal{A}}C_{\mathcal{S}})^D \exp(-\min\{C_{\mathcal{A}}, C_{\mathcal{S}}\}t_{\max}^2))$.

Theorem 3. *[Accuracy of RBSS $\mathcal{Q}$-Value Estimate]. For any $\epsilon > 0$, choose constants $C_{\mathcal{A}}$, $C_{\mathcal{S}}$, λ, δ that satisfy,*

$$\lambda = \epsilon(1-\gamma)^2/5,$$
$$\delta = \lambda/(\mathcal{V}_{\max} D(1-\gamma)^2),$$
$$\delta \geq 3C_{\mathcal{A}}(3C_{\mathcal{A}}C_{\mathcal{S}})^D \exp(-\min\{C_{\mathcal{A}}, C_{\mathcal{S}}\}t_{\max}^2),$$
$$t_{\max} = \frac{\lambda}{3\mathcal{V}_{\max} d_{\infty}^{\max}} - \frac{1}{\sqrt{\min\{C_{\mathcal{A}}, C_{\mathcal{S}}\}}} > 0,$$

Then, the $\mathcal{Q}$-values estimates obtained for all depth D and sampled actions a are near-optimal with probability at least $1-\delta$,

$$\left|\mathcal{Q}_d^*(b_d, a) - \hat{\mathcal{Q}}_d^*(\bar{b}_d, a)\right| \leq \frac{2\lambda}{1-\gamma}.$$

The proof is adapted from Lim et al. (2020) and is detailed in Appendix A.1.2. The key contribution in our proof is to view the ESTIMATE-$\mathcal{V}$ as another SN importance sampling estimator. For this estimator to be properly defined, we need to rely on the representation of reference policy defined in (10). Since (10) is a pushforward measure of $b(s)$ and $b(s)$ is estimated via observation weightings (see ESTIMATE-$\mathcal{Q}$), the importance weightings of $\bar{\pi}$ are simply the observation weightings reweighted by the support of π^{fo}. Then, we can apply the SN concentration bound from Theorem 1 to bound the estimation errors incurred in the ESTIMATE-$\mathcal{V}$ step. Further, the estimation errors from ESTIMATE-$\mathcal{Q}$ can be calculated following the same steps as the presentation of Lim et al. (2020). Hence, the final result is a worst-case union bound that combines all estimation errors from all depths of the tree. Since our ESTIMATE-$\mathcal{V}$ procedure does not exhaustively enumerate through the action space, we can directly work with continuous action spaces where the number of actions we sample is a key parameter that controls the convergence rate.

5 Experiments

We evaluated ROP-RAS3 on seven different long-horizon simulated POMDP problems, systematically analyzing the effects of its respective components on its performance. The simulated testing scenarios include 2D navigation, 3D navigation, multi-robot tag, and manipulation problems. We also demonstrate ROP-RAS3's practicality by deploying it to a Hello-Robot Stretch 3 mobile manipulator. Finally, an ablation study is carried out to understand ROP-RAS3's performance as the reference policy gradually deteriorates to a purely uniform policy. Empirically, we show that the success of ROP-RAS3 is a combination of the new tree search method and reference policies induced from VAMP.

Algorithm 3 Reference-based Sparse Sampling

RBSS($\gamma, \mathscr{G}, C_{\mathcal{A}}, C_{\mathcal{S}}, \bar{b}_0$)

for $i = 1, \ldots, C_{\mathcal{A}}$ **do**
  sample $a_i \sim \bar{\pi}(\cdot \mid \bar{b})$
  $\hat{\mathcal{Q}}_0(\bar{b}, a_i) =$ ESTIMATE-$\mathcal{Q}(\bar{b}, a_i, d)$
end for
$\hat{\pi} \propto \bar{\pi}(a \mid b) \exp(\eta \hat{\mathcal{Q}}_0)$
return $a \sim \hat{\pi}$

ESTIMATE-$\mathcal{V}(\bar{b}, d)$

if $d \geq D$ **then**
  return 0
end if
for $i = 1, \ldots, C_{\mathcal{A}}$ **do**
  sample $a_i \sim \bar{\pi}(\cdot \mid \bar{b})$
  $\nu_i = \sum_{j=1}^{C_{\mathcal{S}}} \mathbb{I}(\pi^{\text{fo}}(s_j) = a_i)\omega_j$
  $\hat{\mathcal{Q}}_d(\bar{b}, a_i) =$ ESTIMATE-$\mathcal{Q}(\bar{b}, a_i, d)$
end for
return $\hat{\mathcal{V}}_d(\bar{b}) = \frac{1}{\eta} \log \left[\frac{\sum_{i=1}^{C_{\mathcal{A}}} \nu_i \exp(\eta \hat{\mathcal{Q}}_d(\bar{b}, a_i))}{\sum_{i=1}^{C_{\mathcal{A}}} \nu_i} \right]$

ESTIMATE-$\mathcal{Q}(\bar{b}, a, d)$

for $s_i \in \bar{b}$ **do**
  $s'_i, o_i, r_i = \mathscr{G}(s_i, a)$
end for
for $i = 1, \ldots, C_{\mathcal{S}}$ **do**
  for $j = 1, \ldots, C_{\mathcal{S}}$ **do**
    $\omega'_j = \omega_j \mathcal{Z}(o_i \mid s'_j, a)$
  end for
  $\bar{b}'_i = (s'_i, \omega_i)$
end for
return $\hat{Q}_d(\bar{b}, a) = \frac{\sum_{i=1}^{C_{\mathcal{S}}} \omega_i r_i + \gamma \text{ESTIMATE-}\mathcal{V}(\bar{b}', d+1)}{\sum_{i=1}^{C_{\mathcal{S}}} \omega_i}$

Table 1. POMDP Summary of Simulation Scenarios. H_{min} denotes the minimum number of steps needed to complete the goal without any uncertainty. Since these scenarios are goal-reaching problems, the minimum planning horizon of the POMDP problem must be higher than the respective H_{min}. Discrete(x) indicates the action space consists of x discrete actions. None corresponds to no observation due to partial observability. Discretized motion path refers to snapping primitive actions to a motion path.

	$\mathcal{S}$	$\mathcal{A}$	$\mathcal{O}$	H_{min}	Uncertainties	PATHTOMACROACTIONS
Light-Dark	$\mathbb{R}^2$	Discrete (4)	$\mathbb{R}^2 \cup \{\text{None}\}$	8	Initial position	Discretised motion path up to a predefined length
Maze2D	$\mathbb{R}^2$	Discrete (4)	$\mathbb{R}^2 \cup \{\text{None}\}$	100	Initial position, Drone transitions Odometry	Discretised motion path up to a predefined length
Random3D	$\mathbb{R}^3$	Discrete (6)	$\mathbb{R}^3 \cup \{\text{None}\}$	40	Initial position Drone transitions Odometry	Discretised motion path up to a predefined length
Multi-Drone	$\mathbb{R}^{15}$	Discrete (24)	$\mathbb{R}^3 \cup \{\text{None}\}$	20	Target locations	Discretised motion path up to a predefined length
Sphere-Search	$\mathbb{R}^{13}$	$\mathbb{R}^7$	$\mathbb{R}^{13} \cup \{\text{None}\}$	50	Target locations Arm transitions	Motion path up to a predefined length
Ray-Detect	$\mathbb{R}^{31}$	$\mathbb{R}^7$	$\mathbb{R}^{31} \cup \{\text{None}\}$	80	Obstacle poses, Arm Transitions.	Motion path up to a predefined length
Shelf-Move	$\mathbb{R}^{35}$	$\mathbb{R}^7$	$\mathbb{R}^{35} \cup \{\text{None}\}$	300	Obstacle poses, Arm Transitions.	Entire motion path

5.1 Simulation Scenarios and Benchmark Methods

The scenarios are described next, while a summary of the corresponding POMDP models is provided in Table 1. More detailed POMDP definitions of each scenario are described in Appendices A.2.1, Table 6 and Table 7.

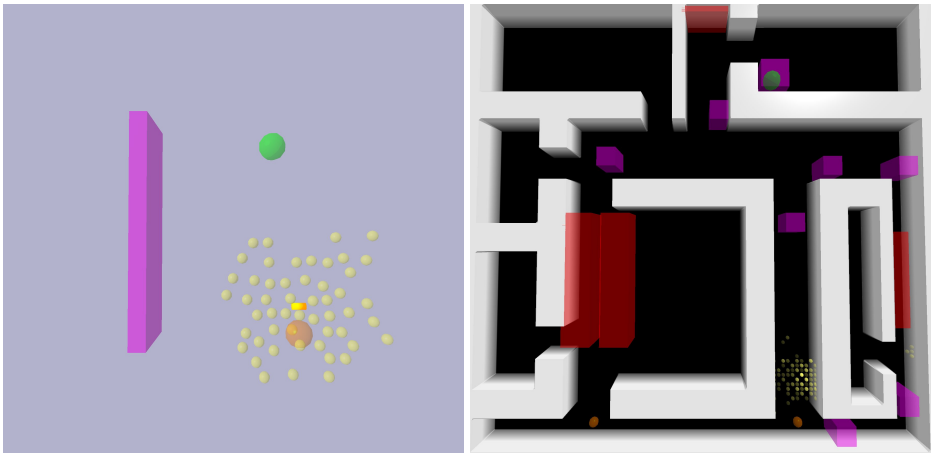
Light Dark (Figure 1a). A variation of the classical Light Dark problem. The agent needs to navigate to a goal region with an initially unknown location. It can only localize in the light stripe.

Maze2D (Figure 1b). This is a long-horizon problem, modified from the discrete 2D Navigation scenario in Kurniawati et al. (2011). A 2-DOF mobile robot must navigate from one of the two potential initial positions to a goal region without entering a danger zone and dodge obstacles. The robot receives position readings with small Gaussian noise inside the landmarks and no observations otherwise. This limited localization capability results in a minimum of 100 primitive actions to reach the goal, as the robot must take detours to localize and avoid danger zones.

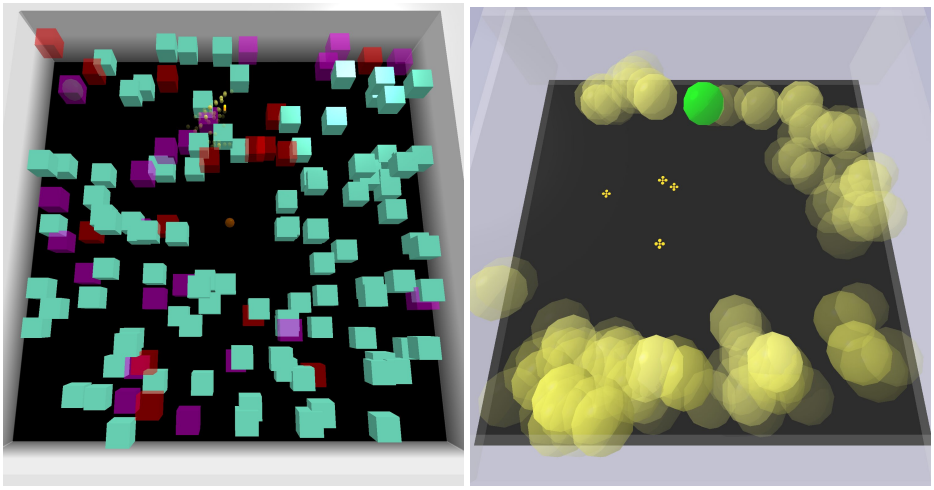
Random3D (Figure 1c). A 3D navigation problem with randomly placed obstacles, landmarks, danger zones and goals. The goals are initialized to be far away from the initial position of the drone. We systematically evaluate ROPRAS3 on environments with progressively increasing obstacle density. The chance that the SBMP fails to find a path due to narrow passages within a given time limit increases with the obstacle density and a POMDP planner needs to consider more diverse macro-actions to find a good motion strategy.

Multi-Drone with A Teleporting Target (Figure 1d). In a 3D open area, four drones need to work together to capture a moving target (in green) whose initial position is not known to the drones. Moreover, the target can *teleport* to the opposite side of the map once it collides with the map boundaries but drones cannot. The target is equipped with the strategy of moving away from the closest drone. This strategy is known to the drones. The drone can only detect the target if they are within a small detection range. And the target is captured as long as one drone is within the capture radius (smaller than detection range).

Sphere-Search (Figure 2a). A 7-DOF manipulator needs to move from its initial configuration to an initially unknown goal location. The robot needs to take a detour to reach a light using its end-effector and observe the exact goal location.

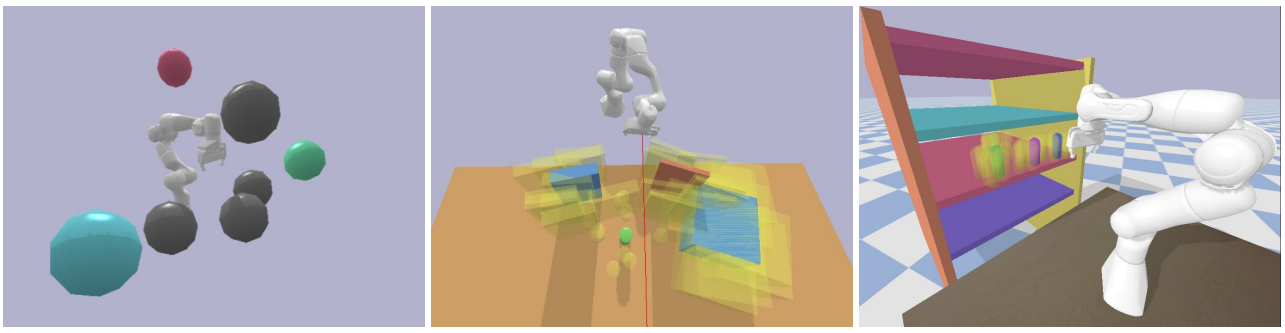


(a) Light Dark **(b)** Maze2D



(c) Random3D **(d)** Multi-Drone Tag

Figure 1. Navigation benchmark environments. All modes of starting locations, if any, are marked in **orange**. Goals are marked in **green**. **Purple** boxes, if any, indicate observation zones. Sampled belief particles are displayed in **Yellow** and the opacity denotes the weight of the particle. **Red** boxes are danger zones. Fixed walls are marked in **gray** and randomly sampled obstacles are marked in **cyan**.



(a) Sphere-Search **(b)** Ray-Detect **(c)** Shelf-Move

Figure 2. Manipulation benchmark environments. Belief particles of observed obstacles are displayed in **yellow**. In Sphere-Search, light is denoted as a **purple** sphere. Goals are marked in **green**. Obstacles are marked in **gray**. In Ray-Detect, obstacles are placed on a table. Rays from the arm is displayed as a **red** straight line. In Shelf-Move, movable obstacles are the cylinders inside the shelf.

Ray-Detect (Figure 2b). A 7-DOF manipulator is tasked to use a fixed line of sight detector mounted at its end effector to scan environments, receives noisy readings of the obstacle positions and find a robust way to reach the target cylinder at the center of the table.

Shelf-Move (Figure 2c). This is the hardest problem that incorporates the difficulties of the other scenarios. A 7-DOF manipulator can move its end-effector close to obstacles to sense their positions. It is tasked to retrieve a target can, placed at the back of the shelf where the front is packed with obstacles, and put the target can at the intended location on the top shelf. The robot can remove the blocking obstacles, but needs to be careful not to place the obstacles at the location reserved for the target.

The following baselines are used for comparisons,

Belief-VAMP (B-VAMP). The planner maintains the belief of the current state of the agent, but only takes actions returned by VAMP's macro-action sampler *without* POMDP planning. It resembles the reference policy used by ROP-RAS3.

Ref-Basic Kim et al. (2023). A reference-based POMDP planner with a uniform sampling reference policy but no VAMP.

POMCP (Silver and Veness 2010). A standard benchmark for online POMDP planning.

R-POMCP. An instantiation of Algorithm 1 using POMCP with macro-actions generated by the same reference policy as ROP-RAS3, but a finite set of macro-actions is sampled for each belief node over which POMCP optimizes. The sample size at each node is set to be roughly equal to that of ROP-RAS3 after progressive widening for fair comparisons.

MAGIC (Lee et al. 2021). A variation of DESPOT (Ye et al. 2017) that uses learnable macro-actions generated via an actor-critic approach.

RMAG (Liang and Kurniawati 2023). DESPOT with macro-action learning boosted by recurrent neural networks.

5.2 Sampling Heuristics

We detail the sampling heuristics (see SAMPLEMACROACTIONSBMP in Algorithm 2) used to create the reference policies for each problem. Biased sampling of informative states is not in itself a new idea, e.g., Kurniawati et al. (2011); Flaspohler et al. (2020); however, using such an idea to construct reference policies is fundamental to the success of ROP-RAS3. Although it is generally impossible to encode (near) optimal solutions as the reference policy, it is much easier to create a reference policy that has compact support over actions that cover the optimal reachable belief space, which in turn makes POMDPs efficient to solve Lee et al. (2007). In our case, such a policy can be obtained by constructing deterministic motion plans to informative states in the world. Informative states refer to states with observations (e.g., landmarks) and rewards (e.g., goals). When these states are not known exactly by the agent, they are modeled as part of the state variable so we can sample these states from the belief particles.

We create two types of sampling heuristics for ablation purposes to understand the importance of the level of information revealed to the agent. The UNIFORM heuristic samples informative states in a uniform manner. It samples the goal state with probability 0.5 and samples the rest of informative states uniformly. The DYNAMIC heuristic varies its sampling strategy according to the belief such that when it is well informed, it is steered towards reaching the goal and when it is less informed, it inclines to gather information. Specifically, it samples the goal with probability $1 - \mathcal{H}(b_t)$, where $\mathcal{H}(b_t)$ is the normalized entropy of the current belief b_t. With probability $\mathcal{H}(b_t)$, it samples the rest of the informative states with probability inversely proportional to the distance of the agent to those states.

An exception is the Multi-Drone problem due to its multi-agent and limited information nature. Since information is only revealed when the drones detect the target, previous heuristics would not be enough to cover the optimal solution as the target can teleport elsewhere but drones cannot. Instead, we create a sampling heuristic that commands the nearest drone to move towards a possible target location sampled from the belief and the rest of the drones spread out uniformly. To further understand how well our method performs, we also conduct ablation studies that gradually remove the information carried by the sampling heuristics in Section 5.6.

5.3 Experimental Setup

For evaluation, all variants of ROP-RAS3 are implemented in Python following Zheng and Tellex (2020). VAMP is implemented in C++ and is used via a Python API. We avoid implementing benchmark methods from scratch for a fairer comparison; the POMCP implementation is from Zheng and Tellex (2020) and the implementations of POMCP, MAGIC, RMAG and Ref-Basic all use the code implemented by their respective authors (Lee et al. 2021; Liang and Kurniawati 2023). PyBullet Coumans and Bai (2016–2021) is used as a visualizer only. For the first four navigation problems, all methods are provided with the same planning time of **1s** in all scenarios with the exception of Light Dark where the planning time is **0.1s**. For manipulation problems, we fix the number of simulations per planning iteration and report the average planning time, as these problems have more computational overhead compared to navigation problems. The temperature parameter η for ROP-RAS3 and Ref-Basic is $\eta = 0.2$. The action progressive widening parameter is set to $\beta_a = 6$ and $\alpha_a = 0.05$. MAGIC and RMAG are trained on a 4070 GPU with data collected in 500,000 runs (∼3 hours of training). The parameters have been tuned to optimize performance for each problem. Other implementation details can be found in Appendix A.2.2.

5.4 Results and Discussions

We ran all methods 30× for each scenario and method, and the results are summarized in Tables 2, 3, 4 and 5. Regardless of success or failure, the "Sims. #" and "Planning Times" columns present the average number of episodes simulated or the time spent in seconds in one planning call, whereas the steps column records the average execution steps for each run. Reward columns indicate the average total reward, and the standard errors are put in brackets. In Table 3, "R.P. Fail %" column indicates the percentage of reference policy failures across 30 runs. Variants of MAGIC and RMAG were run only on Light Dark and Maze2D because they do not naturally extend to high-dimensional motion planning problems (e.g. manipulation and multi-agent tasks). POMCP and Ref-Basic failed on all runs of Random3D and Multi Drones Tag and are therefore excluded from the corresponding tables. Subsequently, we do not expect that these two methods will perform well on manipulation tasks with even longer horizons and higher state dimensions, and hence they are excluded from those experiments. Refer to Figure 6 to Figure 10 in the Appendix for examples of critical time stamps of ROP-RAS3 in performing each tasks.

The results indicate that all variants of ROP-RAS3 substantially outperform all baseline methods in all

Table 2. Results on Light Dark and Maze2D (Red indicates the best result and blue indicates the second best.)

		Light Dark				Maze 2D			
	Sampling Heuristic	**Sims #**	**Succ. %**	**Rewards (std err)**	**Steps**	**Sims #**	**Succ. %**	**Rewards (std err)**	**Steps**
B-VAMP	Uniform	N/A	43.3	38.4 (9.3)	50	N/A	50	-180 (211)	473
B-VAMP	Dynamic	N/A	76	70.2 (7.9)	46	N/A	43	-495 (232)	430
POMCP	N/A	218	56.7	52.5 (9.3)	42	314	0	-80 (0)	800
Ref-Basic	Dynamic	44	50	45.1 (9.3)	49	33	0	-851 (172)	509
MAGIC	N/A	N/A	88	85.0 (1.1)	29	N/A	0	-80 (0)	800
RMAG	N/A	N/A	88	84.8 (1.1)	29	N/A	0	-80 (0)	800
R-POMCP	Uniform	72	80	77.2 (8.1)	30	92	0	-857 (174)	576
R-POMCP	Dynamic	73	76.7	73.9 (8.1)	30	87	0.0	-403 (128)	699
ROP-RAS3	Uniform	21	96.7	94.0 (3.4)	29	198	80	594 (62)	462
ROP-RAS3	Dynamic	3	96.7	93.4 (3.4)	35	43.1	90.0	553 (129)	339

Table 3. Results on Random3D (Red indicates the best result and blue indicates the second best.)
(#1: 100 obstacles, 15 danger zones. #2: 200 obstacles, 10 danger zones.
#3: 300 obstacles, 10 danger zones. #4: 400 obstacles, 5 danger zones. *R.P stands for reference policy*.)

	Sampling Heuristic	**Exp #**	**Sims #**	**Succ. %**	**Rewards (std err)**	**Steps**	**R.P Fail %**	**Exp #**	**Sims #**	**Succ. %**	**Rewards (std err)**	**Steps**	**R.P Fail %**
B-VAMP	Uniform	1	N/A	30	-981 (223)	354	N/A	2	N/A	20	-801(222)	456	N/A
B-VAMP	Dynamic		N/A	6.67	-1237(190)	521	N/A		N/A	6.67	-935(191)	556	N/A
R-POMCP	Uniform		18	13.3	-500(179)	588	N/A		10	13.3	-147(140)	668	N/A
R-POMCP	Dynamic		19	13.3	-553(184)	612	N/A		8	10.0	-263(140)	697	N/A
ROP-RAS3	Uniform		29	63.3	10.6 (213)	318	2.9		16	63	-21.6(212)	364	13
ROP-RAS3	Dynamic		58	66.7	34 (212)	336	2.8		16	53.0	-69.8(220)	400	1
B-VAMP	Uniform	3	N/A	23.3	-535(202)	556	N/A	4	N/A	43.3	-171(198)	520	N/A
B-VAMP	Dynamic		N/A	0	-1062(179)	624	N/A		N/A	13.3	-233(140)	730	N/A
R-POMCP	Uniform		6	10	-191(120)	713	N/A		4	10	-191(122)	713	N/A
R-POMCP	Dynamic		6	10.0	-448(165)	620	N/A		5	6.67	-217(116)	702	N/A
ROP-RAS3	Uniform		9	56.7	-55.7(207)	432	18		7	56.7	73.7(184)	470	23
ROP-RAS3	Dynamic		33	50.0	-371(235)	396	8		15	46.6	183(130)	576	9

Table 4. Results on Multi Drones Tag and Sphere Search (Red indicates the best result and blue indicates the second best.)

	Multi-Drone Tag				Sphere Search			
	Sims #	**Succ. %**	**Rewards (std err)**	**Steps**	**Sims #**	**Succ. %**	**Rewards (std err)**	**Steps**
B-VAMP	N/A	26.7	133 (51.3)	328	N/A	10	65.4 (44.1)	145
R-POMCP	34	66.7	389 (52.6)	251	150	76.7	601 (61.9)	124
ROP-RAS3	47	90	524 (43)	157	150	100	790 (0.31)	104

Table 5. Results on Ray-Detect and Shelf-Move (Red indicates the best result and blue indicates the second best.)

		Ray-Detect (Planning Horizon: 500)					Shelf-Move (Planning Horizon: 1500)				
	Sampling Heuristics	**Sims #**	**Succ. %**	**Rewards (std err)**	**Steps**	**Plan Time (s)**	**Sims #**	**Succ. %**	**Rewards (std err)**	**Steps**	**Plan Time (s)**
B-VAMP	Uniform	N/A	0	-160	800	N/A	N/A	6.67	-243 (38.9)	2968	N/A
B-VAMP	Dynamic	N/A	0	-160	800	N/A	N/A	10	-213 (47.7)	2933	N/A
R-POMCP	Uniform	50	26.7	-66 (28.6)	733	4.4	300	10	-213 (47.7)	2934	35.6
R-POMCP	Dynamic	50	20.0	-93 (24.4)	769	4.5	100	10.0	-214 (47.2)	2939	12.7
ROP-RAS3	Uniform	50	63.3	71.4 (32.3)	597	5.3	300	23.3	-96.7 (67.5)	2835	35.3
ROP-RAS3	Dynamic	50	80.0	137 (27.5)	519	4.6	100	70.0	313 (57.1)	2470	8.9

evaluation scenarios. The improvement provided by ROP-RAS3 is the smallest for Light Dark and Sphere Search (the simplest evaluation scenarios for navigation and manipulations respectively) where all variants of ROP-RAS3 achieved a success rate of up to 28% higher than R-POMCP, MAGIC, and RMAG. The reason is that both of these scenarios require a much lower planning horizon and have less uncertainties compared to other scenarios. In **Light Dark**, this simplicity is indicated by the good performance of methods that do not use macro-actions, such as POMCP and Ref-Basic, and by the good performance of B-VAMP, which reasons with respect to only a sampled state of the belief. In **Sphere Search**, the bimodal uncertainties associated with the goal cause troubles to B-VAMP, however, since the partial observability is fully resolved by reaching the light once and the light is not placed too far away from the manipulator, this problem has a relatively short effective horizon compared to other problems and therefore both R-POMCP and ROP-RAS3 perform well with ROP-RAS3 outperforming R-POMCP by 30%.

In the other scenarios, all variants of ROP-RAS3 improved the success rate of the benchmark methods by many folds. When the scenario requires a much longer planning horizon (e.g., Maze2D, Shelf-Move) or has higher uncertainty and

more complex geometric structures (e.g. Ray-Detect and Shelf-Move), the benefit of ROP-RAS3 increases. By using VAMP, ROP-RAS3 can quickly generate much more diverse macro-actions that capture the geometric features of the problem well. Simultaneously, the reference-based POMDP objective (6) enables ROP-RAS3 to utilize the sampled macro-actions more efficiently than other benchmark methods. The result is that ROP-RAS3 can consistently find the most robust motion path (often requiring longer horizons but has overall greater accumulated rewards) to interact with the environment and reach the goal. Unlike ROP-RAS3, R-POMCP expands all reference policy actions at once when encountering a new belief node in the tree, which means the actions being expanded cannot be adaptive as new belief particles were introduced to the node during planning. By growing the tree in a depth-first-search manner, ROP-RAS3 expands a new action edge at a belief node every time a new particle is added; this way, it benefits the most from the underlying reference policy. As theoretically shown, ROP-RAS3 removes the cumbersome optimization procedures in traditional methods with Monte-Carlo estimations, significantly reducing the number of samples of simulations required to achieve a high success rate in all benchmarks. For the most sophisticated benchmark—Shelf-Move, ROP-RAS3 only requires 100 simulations with under 10s planning time to achieve high success rate (see Table 5). Learning-based methods perform poorly in **Maze2D** due to the difficulty in learning suitable macro-actions with fixed length. In contrast, by using SBMP, ROP-RAS3 is able to better capture the fine motions the robot needs to navigate the geometric features of the environment.

ROP-RAS3 is also robust to performance degradation from the underlying reference policies. We ran 4 variants of **Random3D** with an increasing number of obstacles. The results in Table 3 indicate that ROP-RAS3's performance (especially with uniform random sampling heuristics) does not degrade much even when the reference policy's (*i.e.*, RRT-Connect) failure percentage increases due to increasingly narrower passages in the maze. This is because ROP-RAS3 uses the results of RRT-Connect only to provide a set of alternative sequences of actions to perform. As long as there is sufficient diversity of macro-actions (*i.e.*, sufficient support of the reference policy), the reference-based POMDP planner can converge to a reasonable policy of the POMDP problem fast. This is especially true for the uniform sampling heuristic, as it can provide better diversity which results in more collision free path than dynamic heuristic does and leads to better performances in very cluttered environments.

For the high-dimensional navigation planning problem—**Multi-Drones Tag**, ROP-RAS3 performs at least 4 times better than the rest of the methods as it is the only method that exhibits strategies where drones actively discover and spread out to surround the tag. Both B-VAMP and POMCP cannot easily adapt to the uncertainties from deterministic planning. B-VAMP does not incorporate any uncertainty in the effects of actions and POMCP fixes a set of macro-actions from the reference policy for each belief node based only on a *single* sampled particle. Hence, the expanded set could be insufficient to fully represent the support of the belief, which is a problem that can be further compounded if the reference policy keeps failing.

In **Ray-Detect** (Table 5), ROP-RAS3 is the only method that consistently realizes that the most robust way to reach the cylinder is to use the ray to observe obstacles on the path and swing around the obstacles to avoid getting trapped by cluttered obstacles. Although the belief tree from R-POMCP can reach the same depth as ROP-RAS3, the UCB action selection strategy used by R-POMCP ignores the benefits brought by the sampling heuristics and converges more slowly to find the optimal action. Interestingly, we found out that R-POMCP benefits more from a uniform sampling strategy than a more complex dynamic sampling strategy, as expanding all actions at once requires the underlying sampling strategy to have built-in action diversity, whereas ROP-RAS3 expands action as new particles are introduced to a node, tremendously benefiting from the dynamic sampling strategy.

Shelf-Move is the most sophisticated problem that combines the difficulties of previous problems. Besides requiring very long horizon ($H = 3000$) to deliberately put obstacle cylinders away at carefully planned locations so that the target cylinder can be retrieved and placed at the target location, the scenario itself consists of a non-trivial amount of uncertainties and the clutter of the environment implies occasional failures of finding a deterministic path. Unlike previous problems with a single winning strategy, this problem has many different choices for grasping and placing the cylinders. ROP-RAS3 equipped with the dynamic sampling heuristic is the only one that demonstrates smart grasping and placing of these objects (see the 3rd row of Figure 10). It often identifies that removing two obstacles away at non-target locations will clear a sufficiently large path to retrieve the target cylinder at the back, and place it at the intended location. In case of mistakenly placing an obstacle at the target location, ROP-RAS3 can often find a way to arrange the obstacles correctly for the placement of the target cylinder. Other methods do not exhibit this kind of long horizon thinking and their successful runs were often lucky runs.

5.5 Physical Robot Demonstrations

We deploy ROP-RAS3 on a Hello-Robot Stretch 3 in an uncertain and dynamic environment. In a lab environment, a pedestrian whose position is not exactly known to Stretch moves across the lab with a roughly constant speed. Stretch is tasked to navigate to a goal placed on the other side of the pedestrian. The pedestrian's speed is set such that if going straight, Stretch would most likely run into the pedestrian. The state space has 16 dimensions, of which 13 comes from Stretch, and the remaining 3 are the pedestrian's location as we model the pedestrian as a sphere. We discretize the action space into 45 primitive open loop twist controls that correspond to straight-line and curved motions of the mobile base. The mobile base of Stretch is controlled by twist controllers. We use a simple Euler integrator as the transition function and add noises to it to compensate for the errors between the realized Stretch motion and integrated solution. Stretch can query its IMU sensors to provide actual observations for belief updates. Such an update serves as a filtering step to better localize Stretch and reduce odometry uncertainties. The problem horizon is set to 75, with the planning horizon set to 25. A reward of 800 is provided to

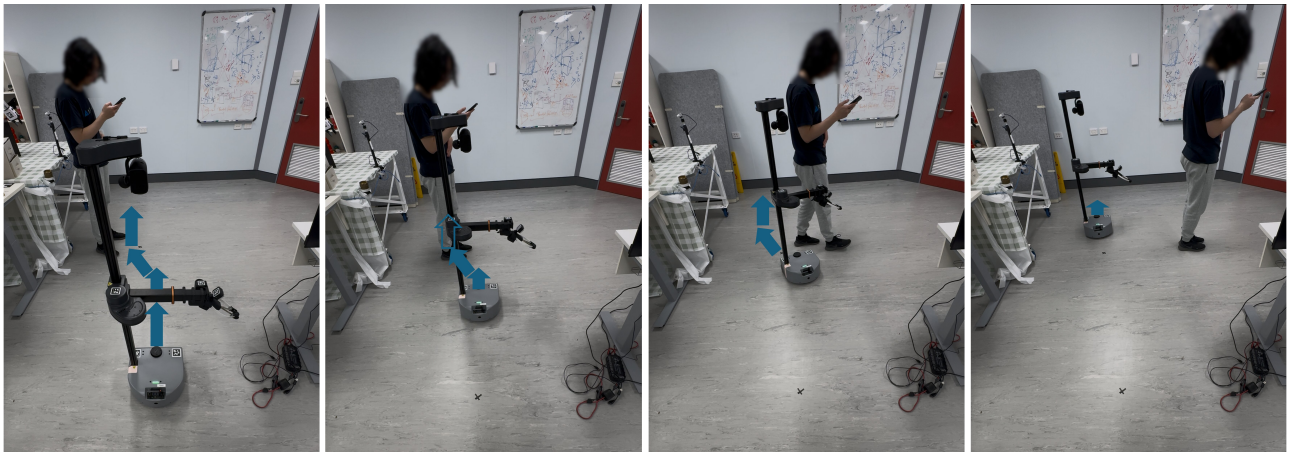


Figure 3. Stretch demonstrations using ROP-RAS3. It smartly navigates around the moving pedestrian and quickly reaches the goal without waiting for the pedestrian or colliding with the environment.

the agent for reaching the goal. A -800 penalty is provided for colliding with the pedestrian. A -20 penalty is given for being too close (within 0.3m) to the pedestrian. Otherwise, a -1 penalty is given for each primitive step taken. We compare ROP-RAS3 with B-VAMP and R-POMCP in a few trials. As shown in Figure 3 and Figure 11, ROP-RAS3 is the only method that demonstrates a consistent smart and robust strategy of taking an efficient detour to go behind the moving pedestrian without colliding with other parts of the environment. B-VAMP in Figure 11 steers straight and bumps the pedestrian as it does not incorporate POMDP planning, whereas R-POMCP took a long detour and ran into the tables. See Appendix A.2.3 for more details on the Stretch implementations.

5.6 Ablation Study

5.6.1 Explorative Reference Policy. One might be concerned that limiting state sampling to only hand-picked information states in the reference policy (see Section 5.2) is too restrictive. Therefore, we also evaluate ROP-RAS3 when these states are sampled in an ϵ-greedy fashion, where with probability ϵ, the sampling heuristic samples from the entire state space, and it samples those information states with $(1-\epsilon)$ probability. When ϵ is 0, it corresponds to the unmodified sampling strategy, and when ϵ is 1, the hand-crafted sampling strategy is purely replaced with uniform random sampling of the state space.

For this ablation study, we test the above sampling strategy on the Maze2D and Multi-Drone Tag scenarios. We also increase the planning time per step from 1s to 3s to allow proper belief space coverage due to the increase of sampling possibilities. For a fair comparison, we also ran the strong baseline, R-POMCP, on the two scenarios for 3s planning time per step.

The results are displayed in Figure 4. As expected, when the reference policy gradually approaches uniform, ROP-RAS3's performance decreases. The reference policy becomes farther away from the deterministic optimal policy with respect to the KL-divergence, as a result the found policy can only perturb the reference policy so much to collect more rewards. However, the performance drop is not linear with respect to ϵ. For instance, in Maze2D, significant amount of drop is seen when ϵ is increased from 0.6 to 1.

Despite the decreasing performance of ROP-RAS3, it generally still performs better than the strong baseline R-POMCP (without any $\epsilon-$explorations when sampling actions). In Maze2D, ROP-RAS3 with pure explorative reference policy (ie., $\epsilon = 1$) still outperforms R-POMCP, with 13% success rate when the policy is generated using ROP-RAS3 and 0% success rate when using R-POMCP. In Multi-Drone scenario, with $\epsilon <= 0.8$, ROP-RAS3 outperforms R-POMCP scenario, indicating the importance of our novel tree search and backup steps.

Finally, this ablation study indicates ROP-RAS3 is less sensitive to the sampling heuristics in Multi-Drone Tag, as its worst success rate is retained at 40%. This is due to the fact that Maze2D is more geometrically complex than Multi-Drone Tag, that is, the chance of sampling the right states to form a robust path is much less in Maze2D than that in Multi-Drone Tag (there are many different ways to surround and capture the target).

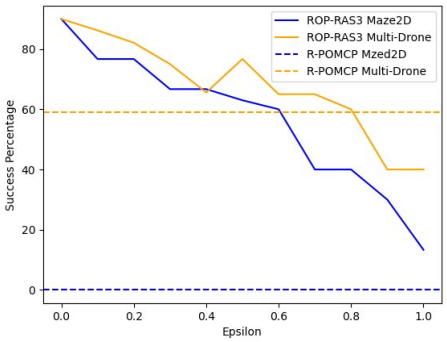


Figure 4. Ablation Study of ROP-RAS3's performances with ϵ-Exploration in Maze2D and Multi-Drone.

5.6.2 Effect of Tree Search Depth. Hyperparameters such as the tree search depth are important to the performance of ROP-RAS3. Under tight computational budgets (e.g., one second of planning), the right tree search depth needs to balance collecting long horizon information and Monte Carlo estimation accuracies. This ablation study perturbs

the tree search depth across two experiments for ROP-RAS3 with the UNIFORM sampling heuristic. Results are shown in Figure 5. We see that performance drops as the tree depth increases or decreases; a good rule-of-thumb we found is to set the tree depth to be roughly the number of steps needed to solve the problem under deterministic settings.

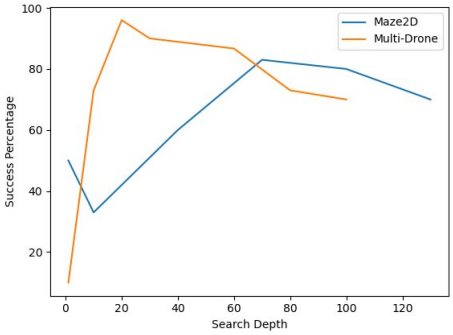


Figure 5. Ablation study of ROP-RAS3's performances with different tree search depths in Maze2D and Multi-Drone.

6 Summary

This paper presents a continuous reference-based POMDP framework to handle large scale robotic problems. The objective of the Reference-based POMDP can be partially solved analytically, resulting in a backup equation that can be estimated with sampling without online numerical optimizations. We then present a new online approximate Reference-based POMDP solver, Reference-Based Online POMDP Planning via Rapid State Space Sampling (ROP-RAS3), which uses VAMP to sample the state space and rapidly generate a large number of macro-actions online. These macro-actions reduce the effective planning horizon required and are adaptive to geometric features of the robot's free space. Since Reference-based POMDP planners use macro-actions only to bias belief-space sampling and do not require exhaustive enumeration of macro-actions, ROP-RAS3 can efficiently exploit many diverse macro-actions to compute good POMDP policies fast. It is shown that the number of belief particles and actions sampled controls the convergence of ROP-RAS3s online planning. Evaluations on various long horizon POMDPs indicate that ROP-RAS3 outperforms state-of-the-art methods by multiple factors. Avenues of future works abounds. Hinted in the ablation study, a poorly selected reference policy could lead to inferior performance, and one may ask what are the characteristics of a reference policy that can guarantee a certain level of performance. Overall, the substantial increase in the capability of online approximate POMDP solvers in long horizon problems brought by ROP-RAS3 enables improved robustness in wide ranges of robotics applications.

7 Acknowledgements

YL, EK, and HK have been partially supported by the ANU Futures Scheme. YL has been partially supported by Australia RTP scholarship. EK has been partially supported by LP200301612. HK has been partially supported by the SmartSat CRC. JAK, ZK and LEK have been supported in part by NSF 2008720, 2336612, and Rice University Funds. WT has been supported by NSF ITR 2127309—CRA CIFellows Project.

References

Azar MG, Gómez V and Kappen H (2011) Dynamic policy programming with function approximation. In: *International Conference on Artificial Intelligence and Statistics*, volume 15. pp. 119–127.

Azar MG, Gómez V and Kappen H (2012) Dynamic policy programming. *Journal of Machine Learning Research* 13: 3207–3245.

Cai P, Luo Y, Hsu D and Lee WS (2021) HyP-DESPOT: a hybrid parallel algorithm for online planning under uncertainty. *International Journal of Robotics Research* 40(2-3): 558–573. DOI:10.1177/0278364920937074.

Coumans E and Bai Y (2016–2021) Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org.

Du S, Kakade S, Lee J, Lovett S, Mahajan G, Sun W and Wang R (2021) Bilinear classes: A structural framework for provable generalization in RL. In: *International Conference on Machine Learning*. PMLR, pp. 2826–2836.

Flaspohler G, Roy NA and Fisher III JW (2020) Belief-dependent macro-action discovery in POMDPs using the value of information. In: Larochelle H, Ranzato M, Hadsell R, Balcan M and Lin H (eds.) *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc., pp. 11108–11118.

He R, Brunskill E and Roy N (2010) PUMA: planning under uncertainty with macro-actions. In: Fox M and Poole D (eds.) *AAAI Conference on Artificial Intelligence*.

Jin C, Yang Z, Wang Z and Jordan MI (2020) Provably efficient reinforcement learning with linear function approximation. In: *Conference on Learning Theory*. PMLR, pp. 2137–2143.

Kaebling L, Littman M and Cassandra A (1998) Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101(1–2): 99–134.

Kavraki LE, Svestka P, Latombe JC and Overmars MH (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation* 12(4): 566–580.

Kim E, Karunanayake Y and Kurniawati H (2023) Reference-based POMDPs. In: *Advances in Neural Information Processing Systems*.

Kim E and Kurniawati H (2025) Partially observable reference policy programming. In: Kwok J (ed.) *International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, pp. 8536–8543. DOI:10.24963/ijcai.2025/949.

Kuffner JJ and LaValle SM (2000) RRT-connect: An efficient approach to single-query path planning. In: *IEEE International Conference on Robotics and Automation*, volume 2. pp. 995–1001.

Kurniawati H (2022) Partially observed Markov decision processes and robotics. *Annual Review of Control, Robotics, and Autonomous Systems* 5: 254–277.

Kurniawati H, Du Y, Hsu D and Lee WS (2011) Motion planning under uncertainty for robotic tasks with long time horizons.

International Journal of Robotics Research 30(3): 308–323.

Kurniawati H and Yadav V (2013) An online POMDP solver for uncertainty planning in dynamic environment. In: *International Symposium on Robotics Research*. pp. 611–629.

LaValle SM (2006) *Planning Algorithms*. Cambridge University Press.

Lee W, Rong N and Hsu D (2007) What makes some POMDP problems easy to approximate? In: Platt J, Koller D, Singer Y and Roweis S (eds.) *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc.

Lee Y, Cai P and Hsu D (2021) MAGIC: Learning Macro-Actions for Online POMDP Planning. In: *Robotics: Science and Systems*. Virtual. DOI:10.15607/RSS.2021.XVII.041.

Liang Y, Kim E, Thomason W, Kingston Z, Kurniawati H and Kavraki LE (2024) Scaling long-horizon online POMDP planning via rapid state space sampling. In: *International Symposium on Robotics Research*.

Liang Y and Kurniawati H (2023) Recurrent macro actions generator for POMDP planning. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 2026–2033. DOI:10.1109/IROS55552.2023.10341759.

Lim MH, Becker TJ, Kochenderfer MJ, Tomlin CJ and Sunberg ZN (2023) Optimality guarantees for particle belief approximation of POMDPs. *Journal of Artificial Intelligence Research* 77: 1591–1636.

Lim MH, Tomlin C and Sunberg ZN (2020) Sparse tree search optimality guarantees in POMDPs with continuous observation spaces. In: Bessiere C (ed.) *International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, pp. 4135–4142. DOI: 10.24963/ijcai.2020/572.

Lim MH, Tomlin CJ and Sunberg ZN (2021) Voronoi progressive widening: Efficient online solvers for continuous state, action, and observation POMDPs. In: *IEEE Conference on Decision and Control (CDC)*. IEEE, pp. 4493–4500.

Papadimitriou CH and Tsitsiklis JN (1987) The complexity of Markov decision processes. *Mathematics of Operations Research* 12(3): 441–450.

Rainforth T, Cornish R, Yang H, Warrington A and Wood F (2018) On nesting monte carlo estimators. In: *International Conference on Machine Learning*. PMLR, pp. 4267–4276.

Shachter RD and Peot MA (1990) Simulation approaches to general probabilistic inference on belief networks. In: *Machine intelligence and pattern recognition*, volume 10. Elsevier, pp. 221–231.

Silver D and Veness J (2010) Monte-Carlo planning in large POMDPs. In: Lafferty J, Williams C, Shawe-Taylor J, Zemel R and Culotta A (eds.) *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc.

Smallwood RD and Sondik EJ (1973) The optimal control of partially observable Markov processes over a finite horizon. *Operations Research* 21(5): 1071–1088.

Sunberg Z and Kochenderfer M (2018) Online algorithms for POMDPs with continuous state, action, and observation spaces. In: *International Conference on Automated Planning and Scheduling*, volume 28. pp. 259–263.

Sundaralingam B, Hari SKS, Fishman A, Garrett C, Van Wyk K, Blukis V, Millane A, Oleynikova H, Handa A, Ramos F, Ratliff N and Fox D (2023) CuRobo: Parallelized collision-free robot motion generation. In: *IEEE International Conference on Robotics and Automation*. pp. 8112–8119. DOI:10.1109/ICRA48891.2023.10160765.

Theocharous G and Kaelbling L (2003) Approximate planning in POMDPs with macro-actions. In: Thrun S, Saul L and Schölkopf B (eds.) *Advances in Neural Information Processing Systems*.

Thomason W, Kingston Z and Kavraki LE (2024) Motions in microseconds via vectorized sampling-based planning. In: *IEEE International Conference on Robotics and Automation*.

Todorov E (2006) Linearly-solvable Markov decision problems. In: Schölkopf B, Platt J and Hoffman T (eds.) *Advances in Neural Information Processing Systems*, volume 19. MIT Press.

Uehara M, Zhang X and Sun W (2021) Representation learning for online and offline RL in low-rank MDPs.

Ye N, Somani A, Hsu D and Lee WS (2017) DESPOT: Online POMDP planning with regularization. *Journal of Artificial Intelligence Research* 58: 231–266.

Zheng K and Tellex S (2020) pomdp_py: A framework to build and solve POMDP problems. In: *ICAPS Workshop on Planning and Robotics (PlanRob)*.

A Appendix

Details of mathematical derivations, proofs and experiments are reported here. We begin with theoretical portion of this work. For convenience, we repeat the theorems again in this section. Notations and assumptions are introduced as they appear. Experimental details are given in the second part of this section.

A.1 Theoretical Analysis

A.1.1 Analytical Solution

Theorem 4. *[Analytical Solution of Referenced-based* POMDP*]. The exact solution of* (6) *is given by,*

$$\mathcal{V}(b) = \frac{1}{\eta} \log \Big[\int_{\mathcal{A}} \bar{\pi}(a \,|\, b) \exp \Big\{ \eta \mathcal{Q}(b,a) \Big\} \mathrm{d}a \Big].$$

Moreover, the exact *solution of the Reference-Based* POMDP *is given by,*

$$\pi^*(a \,|\, b) \propto \bar{\pi}(a \mid b) \exp(\eta \mathcal{Q}(b,a))$$

Proof. At a given belief b, with the constraint that the probability needs to sum to one and positive everywhere, we seek to maximize the following objective,

$$\begin{aligned} \mathcal{L}_b(\pi, \lambda, \alpha) = & \int_{\mathcal{A}} \pi(a \mid b) \mathcal{Q}(b,a) \, \mathrm{d}a - \frac{1}{\eta} \mathrm{KL}(\pi \| \bar{\pi}) \\ & + \lambda \Big(1 - \int_{\mathcal{A}} \pi(a \mid b) \, \mathrm{d}a \Big) - \int_{\mathcal{A}} \alpha(a) \pi(\cdot \mid b) \, \mathrm{d}a \end{aligned} \quad (11)$$

Since $\pi > 0$, the objective is maximized when $\alpha(a) = 0$ for all $a \in \mathcal{A}$. The first order condition applied to the objective implies that a.e.$\forall a \in \mathcal{A}$,

$$\mathcal{Q}(b,a) - \frac{1}{\eta} (\log \pi(a \mid b) - \log \bar{\pi}(a \mid b) + 1) - \lambda = 0$$

Rearrange we obtain that,

$$\pi(a \mid b) = \bar{\pi}(a \mid b) \exp(\eta \mathcal{Q}(b,a) - \eta\lambda - 1) \quad (12)$$

Impose the normalization condition on π and solve for λ,

$$\eta\lambda + 1 = \log \int_{\mathcal{A}} \bar{\pi}(a \mid b) \exp(\eta \mathcal{Q}(b, a)) \, \mathrm{d}a \tag{13}$$

Substitute (13) back to (12), we obtain

$$\pi^*(a \mid b) = \frac{\bar{\pi}(a \mid b) \exp(\eta \mathcal{Q}(b, a))}{\int_{\mathcal{A}} \bar{\pi}(a \mid b) \exp(\eta \mathcal{Q}(b, a)) \, \mathrm{d}\tilde{a}}$$

Substitute the optimal policy back to (11), we have the desired result,

$$\mathcal{V}(b) = \frac{1}{\eta} \log \Big[\int_{\mathcal{A}} \bar{\pi}(a \,|\, b) \exp \Big\{ \eta \mathcal{Q}(b, a) \Big\} \, \mathrm{d}a \Big].$$

□

A.1.2 Convergence Analysis We propose RBSS (Algorithm 3) as a theoretical version of ROP-RAS3 to analyze its online planning convergence. RBSS aims to use action weights and observation weights to compute $\hat{\mathcal{V}}_d$ and $\hat{\mathcal{Q}}_d$, the estimators for the referenced-based $\mathcal{V}$ and $\mathcal{Q}$ values at depth $d \in \{0, \ldots, D-1\}$ of the tree. Given an initial weighted particle set $\bar{b}_0 = \{(s_i, \omega_i)\}_{i=1}^{C_{\mathcal{S}}}$, where states are drawn from the initial belief b_0 and weights $\omega = 1/C_{\mathcal{S}}$, RBSS proceeds to compute the reference-based $\mathcal{Q}$-value after sampling $C_{\mathcal{A}}$ many actions from the reference policy. In ESTIMATE-$\mathcal{Q}$, RBSS has access to a generative model $\mathscr{G}$ to simulate the state transitions, observations and rewards based on the input a. The particle weights can be updated by using the observation weights. The current reward and the next belief is used to compute the $\mathcal{Q}$ by recursively calling for the ESTIMATE-$\mathcal{V}$ to compute the next step expected reward. Similar to ESTIMATE-$\mathcal{Q}$, the action weight ν_i is used as SN weights for estimating the $\mathcal{V}$ expectations, which again depends on the $\mathcal{Q}$-values for the newly sampled actions. The recursion terminates at the specified tree depth D. At tree depth 0, the estimated $\hat{\mathcal{Q}}_0$ values can be used to approximate the optimal reference policy denoted as $\hat{\pi}$ according to (8). The following assumptions are needed for the analysis of RBSS,

1. $\mathcal{S}$, $\mathcal{A}$ and $\mathcal{O}$ are continuous spaces.
2. The Rényi divergence of any target distribution and sampling distribution is bounded above by $d_\infty^{\max} < \infty$.
3. The reward function is Borel and maps to $[0, 1]$.
4. We have access to a black box generator $\mathscr{G}$ to sample next states, rewards and observations given the current state and action. We can also evaluate the observation probability $\mathcal{Z}$.
5. The execution horizon terminates after $D < \infty$ steps.

We relax the discrete action space assumption from Lim et al. (2020) to continuous spaces in assumption 1. The second assumption is needed to invoke Theorem 1. The third assumption is standard, see Jin et al. (2020); Du et al. (2021); Uehara et al. (2021), and it implies that the maximum $\mathcal{V}$ value is $\mathcal{V}_{\max} = 1/(1-\gamma)$. And the last two assumptions are common for online POMDP planners. We emphasize that these assumptions are for theoretical analysis only and are not needed in the experiments. The following theorem shows that the number of particles in a tree node and the number of actions we sample control the convergence of RBSS. Note that the convergence is with respect to the reference-based POMDP objective (7), not the POMDP objective.

Theorem 5. *[Accuracy of RBSS $\mathcal{Q}$-Value Estimate]. Under the aforementioned setup. For any $\epsilon > 0$, choosing constants $C_{\mathcal{A}}$, $C_{\mathcal{S}}$, λ, δ that satisfies,*

$$\lambda = \epsilon(1-\gamma)^2/5,$$

$$\delta = \lambda/(\mathcal{V}_{\max} D(1-\gamma)^2),$$

$$\delta \geq 3C_{\mathcal{A}}(3C_{\mathcal{A}}C_{\mathcal{S}})^D \exp(-\min\{C_{\mathcal{A}}, C_{\mathcal{S}}\} t_{\max}^2),$$

$$t_{\max} = \frac{\lambda}{3\mathcal{V}_{\max} d_\infty^{\max}} - \frac{1}{\sqrt{\min\{C_{\mathcal{A}}, C_{\mathcal{S}}\}}} > 0,$$

Then, the $\mathcal{Q}$-values estimates obtained for all depth D and sampled actions a are near-optimal with probability at least $1-\delta$,

$$\left| \mathcal{Q}_d^*(b_d, a) - \hat{\mathcal{Q}}_d^*(\bar{b}_d, a) \right| \leq \frac{2\lambda}{1-\gamma}.$$

We begin by stating the notation and lemmas used for the proof of the theorem.

Notation. We denote the transition density of state sequence i from the root node to depth d as,

$$\mathcal{T}_{1:d}^i \equiv \prod_{n=1}^{d} \mathcal{T}(s_{n,i} \mid s_{n-1,i}, a_n)$$

And the observation density of state sequence i, observation sequence j from the root node to depth d as,

$$\mathcal{Z}_{1:d}^{i,j} \equiv \prod_{n=1}^{d} \mathcal{Z}(o_{n,j} \mid a_n, s_{n,i})$$

Any absence of indices i, j means that $\{s_n\}$ or $\{o_n\}$ appear as regular variables. When both densities appear together, we use $(\mathcal{T}\mathcal{Z})_{1:d}$ to denote $\mathcal{T}_{1:d}\mathcal{Z}_{1:d}$. We use b_d^i to denote $b_d(s_{d,i})$, $r_{d,i}$ as the reward $R(s_{d,i}, a_d)$ and $\omega_{d,i}$ as the weight of $s_{d,i}$. The following lemma bounds the leaf node $\mathcal{Q}$-value estimations,

Lemma 1. *[Depth $D-1$ $\mathcal{Q}$-value convergence.] For any sampled action a at depth $D-1$, $\hat{\mathcal{Q}}_{D-1}^*(\bar{b}_{D-1}, a)$ is an SN estimator of $\mathcal{Q}_D^*(b_D, a)$. And the following holds with probability at least $1 - 3\exp(-C_{\mathcal{S}} t_{\max}^2)$,*

$$\left| \mathcal{Q}_{D-1}^*(b_{D-1}, a) - \hat{\mathcal{Q}}_{D-1}^*(\bar{b}_{D-1}, a) \right| \leq \lambda$$

Proof. By our finite horizon assumption, at depth $D-1$, the $\mathcal{Q}$-value is simply the expectation of final reward,

$$\mathcal{Q}_{D-1}^*(b_{D-1}, a) = \int_{\mathcal{S}} R(s_{D-1}, a) b_{D-1} \, \mathrm{d}s_{D-1}$$

which is equivalent to the POMDP leaf node case, the rest of the proof follows from Lemma 1 of Lim et al. (2020). □

The standard POMDP backup implies maximizations are required to compute the V-values. Since this step has been done analytically for our reference-based POMDP backup (7), the $\mathcal{V}$-value becomes another round of expectation estimation, which leads to the key contribution of the proof,

Lemma 2. *[Depth $D-1$ $\mathcal{V}$-value convergence]. At depth $D-1$, $\hat{\mathcal{V}}^*_{D-1}(\bar{b}_{D-1})$ is an SN estimator of $\mathcal{V}^*_{D-1}(b_{D-1})$. Conditioned on the event in Lemma 1, the following holds with probability $1-3C_{\mathcal{A}}C_{\mathcal{S}}\exp(-\min\{C_{\mathcal{A}},C_{\mathcal{S}}\}t^2_{\max})$,*

$$\left|\mathcal{V}^*_{D-1}(b_{D-1})-\hat{\mathcal{V}}^*_{D-1}(\bar{b}_{D-1})\right|\leq 2\lambda$$

Proof. At depth $D-1$, consider the absolute difference between the $\mathcal{V}$-value and its estimation,

$$\begin{aligned}&\left|\mathcal{V}^*_{D-1}(b_{D-1})-\hat{\mathcal{V}}^*_{D-1}(\bar{b}_{D-1})\right|\\&\leq\left|\mathbb{E}_{\bar{\pi}(\cdot|b)}\left[\exp(\mathcal{Q}^*_{D-1}(b_{D-1},a))\right]\right.\\&\left.-\frac{\sum_{k=1}^{C_{\mathcal{A}}}\nu_k\exp(\hat{\mathcal{Q}}^*_{D-1}(\bar{b}_{D-1},a_k))}{\sum_{k=1}^{C_{\mathcal{A}}}\nu_k}\right|\equiv(A)\end{aligned}$$

where we have used the definition of $\mathcal{V}$ in (7), the return of ESTIMATE-$\mathcal{V}$ in Algorithm 3, and basic logarithmic properties for the inequality. We then use triangle inequalities to split (A) into two terms,

$$\begin{aligned}(A)\leq&\left|\mathbb{E}_{\bar{\pi}(\cdot|b)}\left[\exp(\mathcal{Q}^*_{D-1}(b_{D-1},a))\right]\right.\\&\left.-\frac{\sum_{k=1}^{C_{\mathcal{A}}}\nu_k\exp(\mathcal{Q}^*_{D-1}(\bar{b}_{D-1},a_k))}{\sum_{k=1}^{C_{\mathcal{A}}}\nu_k}\right|\\&+\left|\frac{\sum_{k=1}^{C_{\mathcal{A}}}\nu_k\exp(\mathcal{Q}^*_{D-1}(\bar{b}_{D-1},a_k))}{\sum_{k=1}^{C_{\mathcal{A}}}\nu_k}\right.\\&\left.-\frac{\sum_{k=1}^{C_{\mathcal{A}}}\nu_k\exp(\hat{\mathcal{Q}}^*_{D-1}(\bar{b}_{D-1},a_k))}{\sum_{k=1}^{C_{\mathcal{A}}}\nu_k}\right|\equiv(B)+(C)\end{aligned}$$

The first difference term, (B), can be seen as importance sampling error and the second term, (C), can be seen as the function estimation error. Define,

$$\Theta_{D-1}(b_{D-1})=\mathbb{E}_{\bar{\pi}(\cdot|b)}\left[\exp(\mathcal{Q}^*_{D-1}(b_{D-1},a))\right]$$

$$\Lambda_{D-1}(\bar{b}_{D-1})=\frac{\sum_{k=1}^{C_{\mathcal{A}}}\nu_k\exp(\mathcal{Q}^*_{D-1}(\bar{b}_{D-1},a_k))}{\sum_{k=1}^{C_{\mathcal{A}}}\nu_k}$$

To bound (B), we firstly show that the $\Lambda_{D-1}(\bar{b})$ is indeed the SN estimator of $\Theta_{D-1}(b)$. By following the recursive belief update (14), the belief term can be expanded as,

$$b_{D-1}(s_{D-1})=\frac{\int_{\mathcal{S}^{D-1}}(\mathcal{TZ})_{1:D-1}b_0\,\mathrm{d}s_{0:D-2}}{\int_{\mathcal{S}^D}(\mathcal{TZ})_{1:D-1}b_0\,\mathrm{d}s_{0:D-1}}\tag{14}$$

Expanding $\Theta_{D-1}(b)$ using (14) and our structural assumption on $\bar{\pi}$ in (10),

$$\begin{aligned}\Theta(b_{D-1})&=\int_{\mathcal{A}}\bar{\pi}(a\mid b_{D-1})\exp(\mathcal{Q}(b_{D-1},a))\,\mathrm{d}a\\&=\frac{\int_{\mathcal{A}}\int_{\mathcal{S}^{D-1}}\exp(\mathcal{Q})\delta_{\pi^{\text{fo}}(s_{D-1})}(a)(\mathcal{TZ})_{1:D-1}b_0\,\mathrm{d}s_{0:D-2}\,\mathrm{d}a}{\int_{\mathcal{S}^D}(a)(\mathcal{TZ})_{1:D-1}b_0\,\mathrm{d}s_{0:D-1}\,\mathrm{d}a}\end{aligned}$$

The density we aim to estimate is $\bar{\pi}(\cdot\mid b_{D-1})$, which is a pushforward measure of b_{D-1}, hence if we have an estimate of b_{D-1}, we can estimate $\bar{\pi}$ easily. The b_{D-1} estimation is done in ESTIMATE-$\mathcal{Q}$ of Algorithm 3. Importantly, recall from Lim et al. (2020), let $\mathcal{P}^{D-1}$ denote the normalized measure incorporating the observation sequence j on top of the state sequence i and $\mathcal{F}^{D-1}$ is the probability of the state sequence.

$$\mathcal{P}^{D-1}=\mathcal{P}^{D-1}_{\{o_n\}_j}(\{s_n\}_i)=\frac{\mathcal{T}^i_{1:D-1}\mathcal{Z}^{i,j}_{1:D-1}b^i_0}{\int_{\mathcal{S}^D}\mathcal{T}_{1:D-1}\mathcal{Z}^j_{1:D-1}b_0\,\mathrm{d}s_{0:D-1}}$$

$$\mathcal{F}^{D-1}=\mathcal{Q}^{D-1}(\{s_n\}_i)=\mathcal{T}^i_{1:d}b^i_0$$

Then, the belief importance weight is given by,

$$\omega_{\mathcal{P}^{D-1}/\mathcal{F}^{D-1}}(\{s_n\}_i)=\frac{\mathcal{Z}^{i,j}_{1:D-1}}{\int_{\mathcal{S}^{D-1}}\mathcal{T}_{1:D-1}\mathcal{Z}^j_{1:D-1}b_0\,\mathrm{d}s_{0:D-1}}\tag{15}$$

The action importance weight is given by,

$$\nu_{\mathcal{P}^{D-1}/\mathcal{F}^{D-1}}(a_k)=\sum_{i=1}^{C_{\mathcal{S}}}\mathbb{I}_{\pi^{\text{fo}}(s_n,i)}(a_k)\omega_{\mathcal{P}^{D-1}/\mathcal{F}^{D-1}}(\{s_n\}_i)\tag{16}$$

Fixing an observation sequence $\{o_j\}$, the weight of the actions sampled at depth $D-1$ is given by the observation weight reweighted by the fully observable policy (see Algorithm 3),

$$\begin{aligned}\nu_{D-1,k}&=\sum_{i=1}^{C_{\mathcal{S}}}\mathbb{I}_{\pi^{\text{fo}}(s_{D-1},i)}(a_k)\omega_{D-1,i}\\&\propto\sum_{i=1}^{C_{\mathcal{S}}}\mathbb{I}_{\pi^{\text{fo}}(s_{D-1},i)}(a_k)\mathcal{Z}^{i,j}_{1:D-1}\end{aligned}\tag{17}$$

The last equation come from the recursive update of the belief weights in Algorithm 3. The estimator $\Lambda_{D-1}(\bar{b})$ can be re-written as,

$$\Lambda_{D-1}(\bar{b})=\frac{\sum_{k=1}^{C_{\mathcal{A}}}\sum_{i=1}^{C_{\mathcal{S}}}\mathbb{I}_{\pi^{\text{fo}}(s_{D-1},i)}(a_k)\mathcal{Z}^{i,j}_{1:D-1}\exp(\eta\hat{\mathcal{Q}}_{D-1}(\bar{b},a_k))}{\sum_{k=1}^{C_{\mathcal{A}}}\sum_{i=1}^{C_{\mathcal{S}}}\mathbb{I}_{\pi^{\text{fo}}(s_{D-1},i)}(a_k)\mathcal{Z}^{i,j}_{1:D-1}}$$

Using (15), (16) and (17), we get

$$\begin{aligned}\Lambda_{D-1}(\bar{b})&=\frac{\sum_{k=1}^{C_{\mathcal{A}}}\nu_{\mathcal{P}^{D-1}/\mathcal{F}^{D-1}}(a_k)\exp(\eta\hat{\mathcal{Q}}_{D-1}(\bar{b}_{D-1},a_k))}{\sum_{k=1}^{C_{\mathcal{A}}}\nu_{\mathcal{P}^{D-1}/\mathcal{F}^{D-1}}(a_k)}\\&=\sum_{k=1}^{C_{\mathcal{A}}}\tilde{\nu}_{\mathcal{P}^{D-1}/\mathcal{F}^{D-1}}\exp(\eta\hat{\mathcal{Q}}_d(\bar{b}_{D-1},a_k))\end{aligned}$$

Our assumption that R is a bounded Borel function implies that $\exp(\mathcal{Q}_{D-1})$ is also a bounded Borel function. Since the states are i.i.d sequences drawn from $\mathcal{F}^{D-1}$, we can apply the SN concentration bound from Theorem 1 to bound the importance sampling error by λ with probability at least $1-3\exp(-C_{\mathcal{A}}t^2_{\max})$.

Next the $\mathcal{Q}$ function estimation error (C) is bounded by our conditioning on Lemma 1. With a union bound on the two events, we obtain the final result. □

Recursively, we can then bound the $\mathcal{Q}$-value estimate at arbitrary depth d.

Lemma 3. *[Any depth d $\mathcal{Q}$-value convergence.] At any depth $d = \{0, \dots, D-1\}$, $\hat{\mathcal{Q}}^*_d(\bar{b}_d, a)$ is an SN estimator of $\mathcal{Q}^*_d(b_d, a)$ for any action $a \in \mathcal{A}$. And the following holds with probability at least $1 - 3C_{\mathcal{A}}(3C_{\mathcal{A}}C_{\mathcal{S}})^D \exp(-\min\{C_{\mathcal{A}}, C_{\mathcal{S}}\}t^2_{\max})$,*

$$\left|\mathcal{Q}^*_d(b_d, a) - \hat{\mathcal{Q}}^*_d(\bar{b}, a)\right| \leq \alpha_d \tag{18}$$

$$\alpha_d \equiv 2\lambda + \gamma\alpha_{d+1}; \alpha_{D-1} = \lambda$$

Proof. We pick $C_{\mathcal{A}}$ and $C_{\mathcal{S}}$ such that $\min\{C_{\mathcal{A}}, C_{\mathcal{S}}\} > (3\mathcal{V}_{\max}d^{\max}_{\infty}/\lambda)^2$ to satisfy $t_{\max} > 0$ which ensures the probability holds at any depth d and a. The $\mathcal{Q}$-value estimate defined in ESTIMATE-$\mathcal{Q}$ is given as,

$$\hat{\mathcal{Q}}^*_d(\bar{b}_d, a) = \frac{\sum_{i=1}^{C_{\mathcal{S}}} \omega_{d,i}\left(r_{d,i} + \gamma\hat{\mathcal{V}}^*_{d+1}(\overline{b_d a o_i})\right)}{\sum_{i=1}^{C_{\mathcal{S}}} \omega_{d,i}}$$

We are interested in bounding the gap between the estimator and the actual $\mathcal{Q}^*$. At the leaf node, the gap is bounded by λ given by Lemma 1. Hence $\alpha_{D-1} = \lambda$. Next, the $\mathcal{Q}$ estimation error for any depth d can be separated into two components using the triangle inequality,

$$\begin{aligned}
&\left|\mathcal{Q}^*_d(b_d, a) - \hat{\mathcal{Q}}^*_d(\bar{b}_d, a)\right| \\
&\quad\leq \underbrace{\left|\mathbb{E}\left[R(s_d, a) \mid b_d\right] - \frac{\sum_{i=1}^{C_{\mathcal{S}}} \omega_{d,i} r_{d,i}}{\sum_{i=1}^{C_{\mathcal{S}}} \omega_{d,i}}\right|}_{(D)} \\
&+ \gamma \underbrace{\left|\exp\left[\mathcal{V}^*_{d+1}(bao) \mid b_d\right] - \frac{\sum_{i=1}^{C_{\mathcal{S}}} \omega_{d,i}\hat{\mathcal{V}}^*_{d+1}(\overline{b_d a o_i})}{\sum_{i=1}^{C_{\mathcal{S}}} \omega_{d,i}}\right|}_{(E)}
\end{aligned}$$

Since the form of $\mathcal{Q}$-value is the same as POMDP Q-values, using results from Lim et al. (2020), we can bound (D) above by $\frac{R_{\max}}{3\mathcal{V}_{\max}}$. And by noticing that the proof of Lemma 2 can be recursively applied to all depths d when the $\mathcal{Q}_d$ at the same depth is bounded by results from Lim et al. (2020), we can then bound (E) from above by $\lambda + \frac{1}{3}\lambda + \frac{2}{3\gamma}\lambda + \alpha_{d+1}$. Putting everything together, we have $(D) + (E) \leq 2\lambda + \gamma\alpha_{d+1}$, which is α_d stated in this Lemma. To compute the worst case probability accounting for all events happened in the past, we multiply the base probability by $(4C_{\mathcal{A}}C_{\mathcal{S}})^D$, as we want the function estimations to be within their thresholds at all depths from d, where 4 is the number of times we applied the SN concentration bound at a given depth (1 time in $\mathcal{V}$ estimation and 3 times in $\mathcal{Q}$ estimations). We then multiply the factors by another $3C_{\mathcal{A}}$ to account for the root node $\mathcal{Q}$-value estimations. Therefore, equation (18) holds at all depth $d = 1, \dots, D-1$ with probability at least $1 - 3C_{\mathcal{A}}(4C_{\mathcal{A}}C_{\mathcal{S}})^D \exp(-\min\{C_{\mathcal{A}}, C_{\mathcal{S}}\}t^2_{\max})$. □

Proof. (Theorem 5). We pick constants $\lambda, C_{\mathcal{A}}, C_{\mathcal{S}}, \delta$ and densities $\mathcal{T}, \mathcal{Z}, \pi^{\text{fo}}, b_0$ that satisfies the requirements in Theorem 5. Then with probability $1 - \delta$ through Lemmas 1, 2 and 3, we have

$$\left|\mathcal{Q}^*_d(b_d, a) - \hat{\mathcal{Q}}^*_d(\bar{b}_d, a)\right| \leq \alpha_0 \leq 2\sum_{d=0}^{D-1} \gamma^d\lambda \leq \frac{2\lambda}{1-\gamma}$$

□

A.2 Experiment Details

A.2.1 Simulation Details. The POMDP definition of each benchmark scenarios is summarized into Table 6 and Table 7. In Table 6, we use S_p to denote the joint space of the Franka Panda arm and H to denote the maximum horizons we set for each problem. In Table 7, the probability distributions are noises that i.i.d added to each dimension of the state or observations. In sphere search, $(0.5, 0.76, 0.4)$ and $(0.5, -0.76, 0.4)$ denotes the two possible target locations. For Light-Dark, Random3D, Ray-Detect and Shelf-Move, the initial belief is the true state with Gaussian noises. In Maze2D, the drone can spawn at either entry point, hence the initial belief is a bimodal Dirac delta distribution. In Sphere-Search, the initial belief includes both adding Gaussian noises to the initial manipulator configuration and a bimodal Dirac delta distribution for the possible goal locations. In Multi-Drone, the drone locations are fully observable at all times, but the target location is initially uniformly distributed across the entire workspace.

We also show critical time stamps of ROP-RAS3 in each experiment conducted, demonstrating its long horizon thinking and careful planning capabilities under uncertainty. See Figure 6 to Figure 10.

A.2.2 Implementation Details. When implementing ROP-RAS3, it is important to keep a good balance between the number of simulations ROP-RAS3 computes and the amount of time it takes to compute. In complex scenarios, we found at least 30 to 50 simulations were needed to find good solutions. Since the number of simulations is low, we also do not keep too many belief particles, thus reducing belief update time (see (14)). However, small number of particles often lead to particle deprivation, hence particle reinvigoration was needed to inject a small amount of noise to the beliefs, this is done after the belief update. Another important parameters to tune is the search depth D. Although ROP-RAS3 can theoretically handle longer horizons, it is not desired to set D to be too large as otherwise it compromises the number of simulations or the time it takes to complete one planning. We found a good heuristic is to set D to be roughly the number of steps it takes to complete the task in the fully observable deterministic case. In all problem scenarios considered in this work, it is easy to define a good metric between observations (e.g. whether same objects observed are close to each other in terms of Euclidean distance), but harder to find a good metric for the actions (especially in manipulation tasks), we therefore only used action progressive widening in our implementation and replace observation widening with binning.

A.2.3 Stretch Details. Since VAMP is a kinematic planner but the Stretch 3 has a non-holonomic base, we have to convert the VAMP path to a trajectory the Stretch 3 can follow. We do this within the reference policy by converting a path to a sequence of actions (a macro-action). Each action in the macro-action sequence is a twist (linear and angular velocity) for the base held over 0.7 seconds. We set the amount of time for each action to be held heuristically based on the task and environment. We can therefore send the result from the ROP-RAS3 planner (a sequence of twists) directly to the robot for open-loop control, alternating between planning and execution. To obtain the macro-action from a VAMP

path, we choose the next action in our macro-action to be the action (from 45 possible) that minimizes some error from a lookahead point. The error in our experiments is a weighted sum of 1.0 for the distance to the lookahead point and 0.5 for the heading difference from following the straight line path. This cost was also set heuristically based on how well the trajectory tracked the path. The actions were also chosen heuristically based on obtaining a diversity of possible speeds for natural behavior. The actions consist of three linear speeds ($0.1m/s$, $0.2m/s$, $0.3m/s$) forward and backward along with angular velocities spaced by $0.1rad/s$ between $-0.5rad/s$ and $0.5rad/s$. Note that the Stretch 3 controller for tracking linear and angular velocities follows trapezoidal motion profiles with a set (linear) acceleration of $0.12m/s^2$ for each wheel and a maximum absolute velocity of $0.3m/s$. Therefore, we must integrate over each action to determine how the robot transitions based on the current wheel velocities. Although values and models are chosen in a crude manner, part of the benefit of POMDP planning is to take these uncertainties and errors into account and find a robust plan to execute.

A.2.4 Connections to First Work. Improvements have been made to ROP-RAS3 compared to our first work Liang et al. (2024). With our theoretical insights and double progressive widening techniques, ROP-RAS3 can now handle fully continuous POMDPs. Meanwhile, the original backup equation adjusts the estimations of the $\mathcal{V}$-values by subtracting away the old estimations for $\mathcal{Q}$-values and plug the new estimated $\mathcal{Q}$-values in. Our theory now indicates the backup can be done by simply applying Monte-Carlo estimators at each nested level. Further information on nested Monte-Carlo estimations can be found from Rainforth et al. (2018).

Table 6. Summary of POMDP Spaces and Discounts of All Benchmark Scenarios

	State Space $\mathcal{S}$	Action Space $\mathcal{A}$	Observation Space $\mathcal{O}$	γ
Light-Dark	$[-4, 4]^2$m	$\{(0.5, 0), (-0.5, 0)$ $(0, 0.5), (0, -0.5)\}$m	$[-4, 4]^2$m $\times$ {None}	0.99
Maze2D	$[-25, 25]^2$m	$\{(0.5, 0), (-0.5, 0)$ $(0, 0.5), (0, -0.5)\}$m	$[-25, 25]^2$m $\times$ {None}	0.999
Random3D	$[-25, 25]^3$m	$\{(0.5, 0, 0), (-0.5, 0, 0)$ $(0, 0.5, 0), (0, -0.5, 0),$ $(0, 0, 0.5), (0, 0, -0, 5)\}$m	$[-25, 25]^3$m $\times$ {None}	0.999
Multi-Drone	$([-15, 15]^2 \times [-2, 2]\text{m})^5$	Same as Random3D but applied to all drones	$([-15, 15]^2\text{m} \times [-2, 2]\text{m}) \times$ {None}	0.99
Sphere-Search	S^7_prad$\times$ $\{(0.5, 0.76, 0.4),$ $(0.5, -0.76, 0.4)\}$m	S^7_prad	S^7_prad$\times$ $\{(0.5, 0.76, 0.4), (0.5, -0.76, 0.4)\}m\times$ {None}	0.99
Ray-Detect	S^7_prad$\times$ $([-1, 1]^3 m \times S^3\text{rad})^4$	S^7_prad	S^7_prad $\times [-1, 1]^3 m \times S^3 \times$ {None}	0.999
Shelf-Move	S^7_prad $\times ([-1, 1]^3 m)^4$	S^7_prad	S^7_prad $\times ([-1, 1]^3 m)^4 \times$ {None}	0.9999

Table 7. Summary of POMDP Functions and Horizons of All Benchmarks

	$\mathcal{T}$	$\mathcal{Z}$	R	H
Light-Dark	Deterministic	In Light: $\mathcal{N}(0, 0.1)$m Else: None	Step: -0.1 Goal: 100	60
Maze2D	20% execute a wrong action	In Light: $\mathcal{N}(0, 0.5)$m Else: None	Step: -0.1 Goal: 800 Fail: -2000	800
Random3D	20% execute a wrong action	In Light: $\mathcal{N}(0, 0.5)$m Else: None	Step: -0.1 Goal: 800 Fail: -2000	800
Multi-Drone	Deterministic Target can teleport Drones cannot	In range: $\mathcal{N}(0, 0.5)$m Else: None	Step: -0.1 Goal: 600	400
Sphere-Search	$\mathcal{N}(0, 0.02)$rad	Joints are observable with $\mathcal{N}(0, 0.02)$ rad noises. In light: target fully observable Else: None.	Step: -0.1 Goal: -800 Fail: 800	150
Ray-Detect	$\mathcal{N}(0, 0.02)$rad	Joints are observable with $\mathcal{N}(0, 0.02)$ rad noises. If detect: object position + $\mathcal{U}(-0.02, 0.02)$m object orientation + $\mathcal{U}(-0.04, 0.04)$rad Else: None	Step: -0.2 Goal: 300	800
Shelf-Move	$\mathcal{N}(0, 0.001)$rad	Joints are observable with $\mathcal{N}(0, 0.02)$ rad noises. If detect: object position with $\mathcal{U}(-0.04, 0.04)$ m Else: None	Step: -0.1 Goal: 800	3000

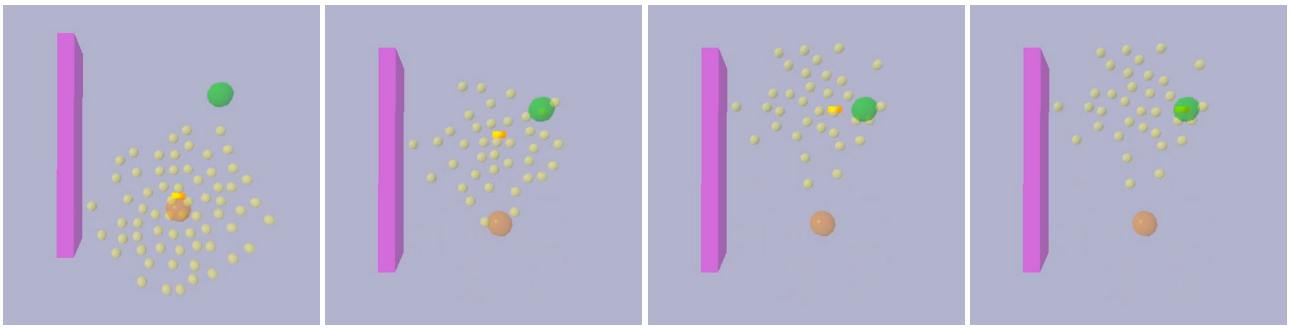


Figure 6. Light Dark is the easiest problem and the agent can quickly navigate to the goal.

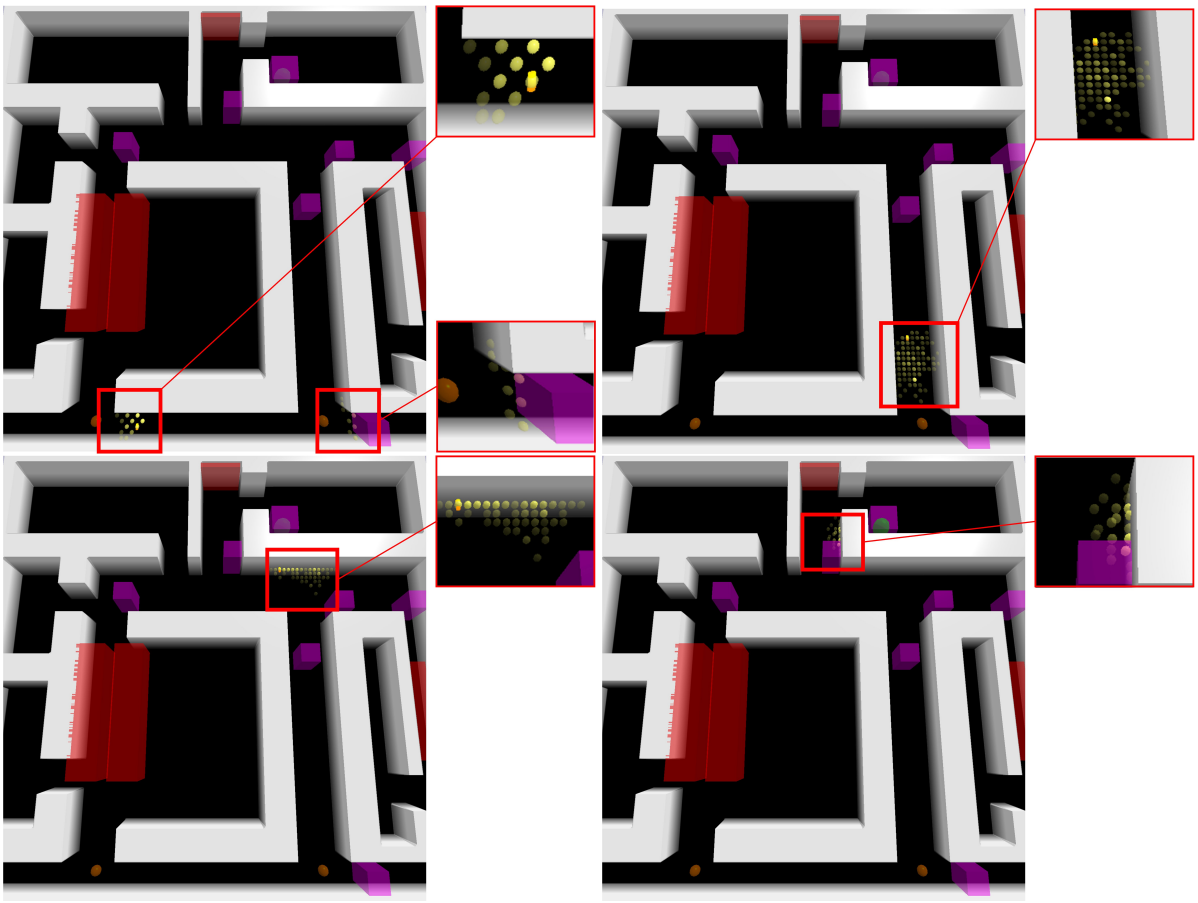


Figure 7. In Maze, the agent avoids the corridors with danger zones and starts going towards right first to localize. As it navigates to the middle corridor, it needs to carefully pick landmarks to receive observations of itself. Sometimes, the agent uses walls to align all the beliefs on one side as a localization mechanism, and eventually managed to reach the goal.

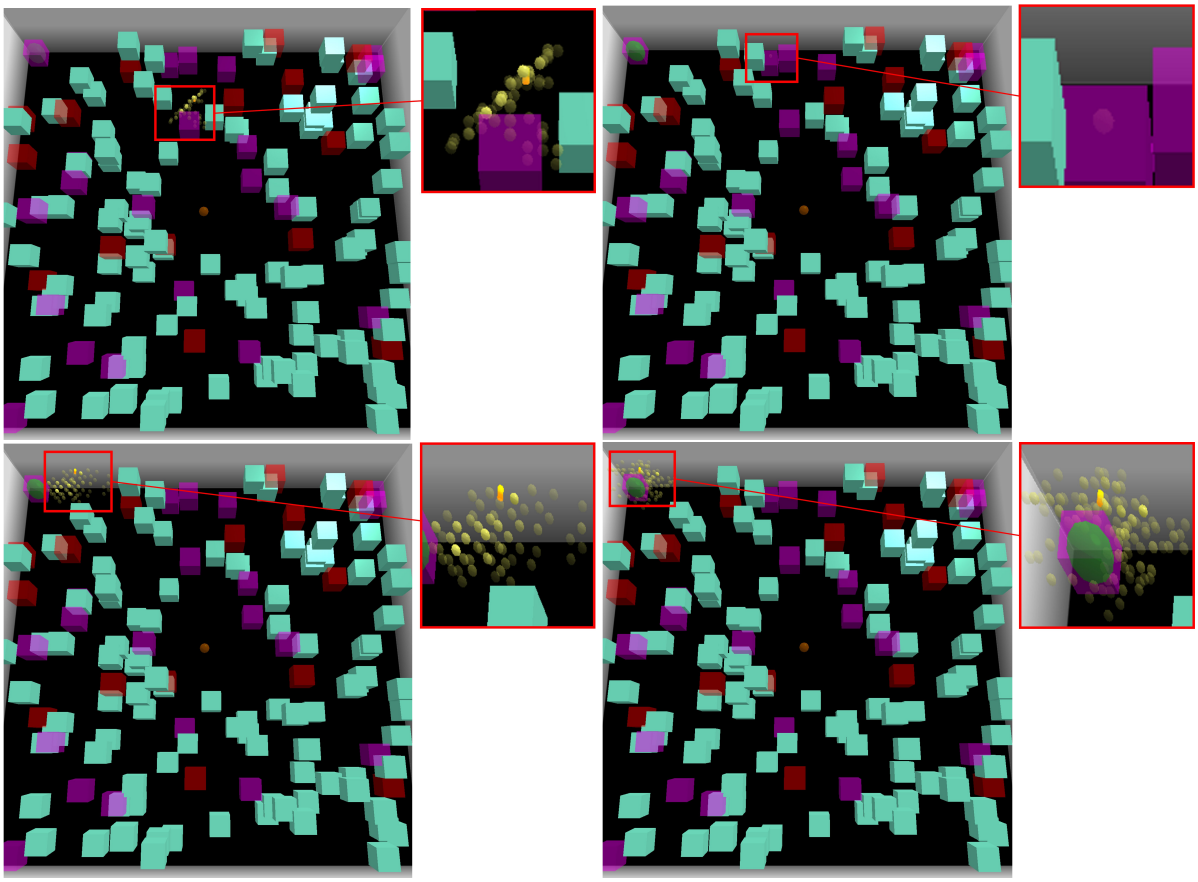


Figure 8. In Random3D, the agent firstly went to the closest landmark on the way to the goal. It then sticks to the wall as it's a robust path, belief particles can all aligned to the wall, avoiding the danger zones and efficiently navigate to the goal.

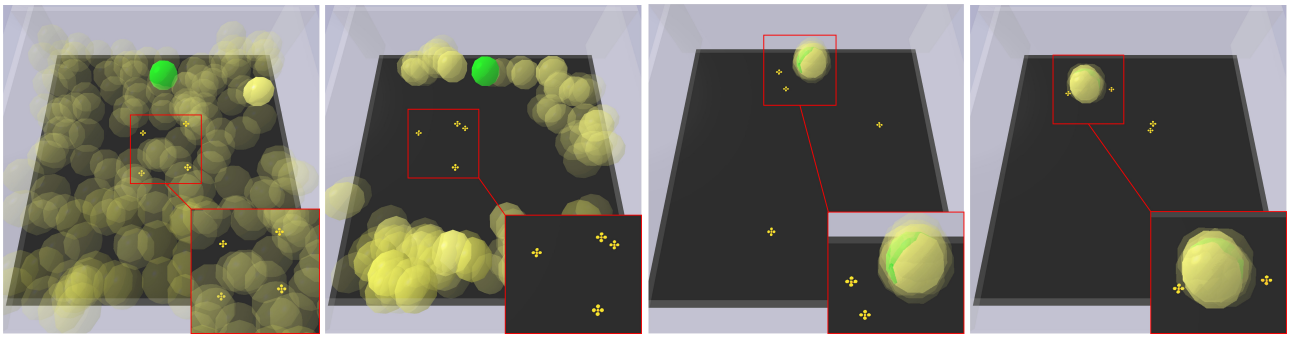


Figure 9. In Multi-Drone, ROP-RAS3 commands the drones to spread out to detect the target. Then one drone aims to chase a downward moving target to force it to teleport to the other side where another agent has been waiting to capture the target.

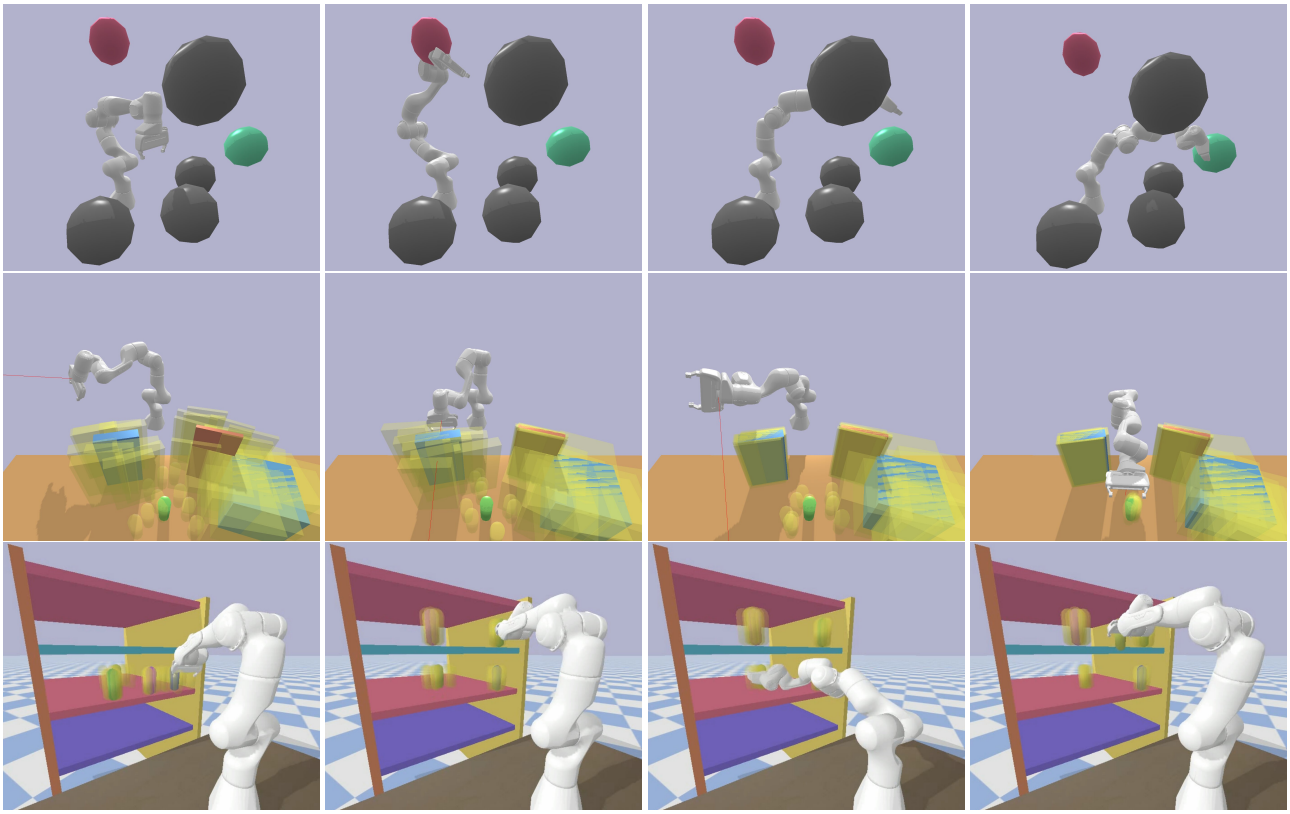


Figure 10. Behaviors of ROP-RAS3 across all manipulation tasks. In Sphere Search (1st row), the manipulator firstly reaches the light (purple) to receive an observation on where the sphere (green) is spawned. Then it navigates directly towards the sphere, avoiding the obstacles (gray). In Ray Detect (2nd row), the manipulator spends extra steps to detect two important obstacles colored in blue and brown. As the uncertainties are reduced (sampled particles are displayed in yellow), the manipulator is able to navigate around the obstacles to reach the target. In Shelf Move (3rd row), the manipulator firstly moves its end effectors around to sense the world, then it place two important cylinders away to the top shelf, leaving the middle position at the top shelf empty for the target. It then grasps the target and placed it at the intended location.

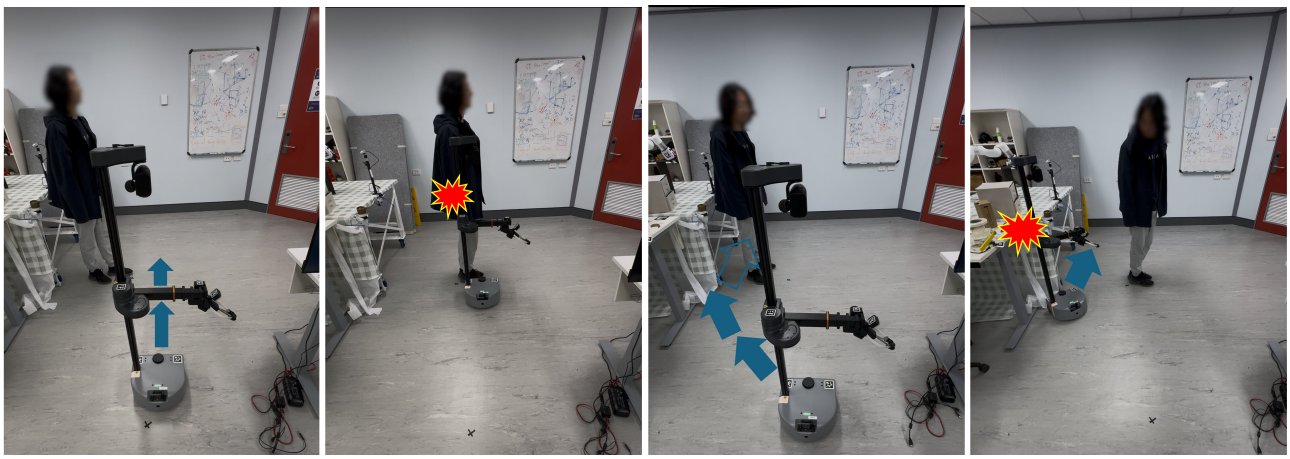


Figure 11. Stretch demonstrations for B-VAMP (first two pictures) and R-POMCP (last two pictures). B-VAMP moves forward and collide with the moving pedestrian due to lack of POMDP planning and R-POMCP tries to avoid the person but went too far and collided with the environment.