# Online Planning in Continuous POMDPs with Open-Loop Information-Gathering Plans

Kris Hauser
School of Informatics and Computing, Indiana University Bloomington
hauserk@indiana.edu

***Abstract*—This paper studies the convergence properties of a receding-horizon information-gathering strategy used in the recently presented RBSR planner for continuous POMDPs. The planner uses a combination of randomized exploration, particle filtering, and goal-seeking heuristic policies to achieve scalability to high-dimensional continuous spaces. We show that convergence is ensured in a subclass of problems where information gain rate exceeds the rate of information loss through process noise. Because these rates are not defined myopically RBSR is able to perform long open-loop information-gathering plans. The technique is demonstrated on a variety of discrete planning benchmarks as well as target-finding and localization problems in up to 7D continuous state spaces.**

## I. Introduction

Although state-of-the-art planners for discrete partially-observable Markov decision processes (POMDPs) have made tremendous recent advances, POMDPs in continuous, high-dimensional state spaces are still far out of the realm of tractability. The primary benefit of the POMDP formulation is that *active sensing* actions — information-gathering actions taken in the pursuit of a goal — are natural byproducts of optimal policies. But many POMDP problems can be solved effectively without invoking the full machinery of optimality, e.g., by applying heurisics based on passive sensing or explicit information gathering rewards. Therefore it is highly valuable to identify scalable, general heuristics that perform well in wide subclasses of POMDPs.

In prior work [7] we presented a Randomized Belief-Space Replanning (RBSR) technique that addressed continuous POMDPs using a online approach [22], where each step maximizes expected return over a reduced space of policies that consist of open-loop actions followed by the closed-loop QMDP policy (Figure 1). These components are complementary; QMDP provides excellent goal guidance under low uncertainty [16], while the open-loop exploration enables active sensing actions to be taken as long as they aid progress toward the goal. Though the policy space is relatively limited, the system exhibits rich behavior by adapting to incoming information through online replanning. A major benefit of this technique is that it can be made scalable to high-dimensional continuous spaces and large exploration depths using a combination of randomized exploration, particle filtering, and Monte-Carlo estimation techniques.

It was observed that RBSR can successfully solve problems that require long information gathering plans, but is less successful under high process noise or when conditional information gathering plans are needed in order make measurable progress. In this paper we sharpen these observations. In particular we prove that there exists a sequence of open-loop actions that makes progress toward the goal provided that a certain measure of information gain rate exceeds the measure of information loss rate (from process noise). We use this result to explain how the use of randomization and approximation in RBSR affects convergence.

Experiments demonstrate that RBSR performs significantly better than QMDP, and surprisingly, sometimes nearly as well as current state-of-the-art offline solvers on discrete planning benchmarks. It is also shown to solve continuous target-finding and localization problems in up to 7D state spaces.

## II. Related Work

Optimal planning in partially-observable problems is extremely computationally complex and is generally considered intractable even for small discrete state spaces [17]. Approximate planning in discrete spaces is a field of active research, yielding several techniques based on the point-based algorithms devised by Kearns et al [12] and Pineau et al (2003) [18]. For example, the SARSOP algorithm developed by Kurniawati et al (2008) has solved problems with thousands of discrete states in seconds [14].

One approach to continuous problems is to discretize state/action/observation spaces. But because of the "curse of dimensionality", any regular discretization of a high-dimensional space will requires an intractably large number of states. Porta et al (2006) has made progress in extending point-based value iteration to the continuous setting by representing belief states as particles or mixtures of Gaussians [20]. Thrun (2000) presented a technique that also works with continuous spaces by combining particle filtering with reinforcement learning on belief states [25]. For both of these methods, the need to approximate the value function over the infinite-dimensional belief space (either using alpha-vector or Q-value representations, respectively) comes at a high computational and memory expense. We use similar representations, but because we use replanning to avoid explicit policy representation, our approach sacrifices near-optimality for reduction in computational expense.

Several recently developed algorithms attempt to address continuous spaces by leveraging the success of probabilistic roadmaps (PRMs) in motion planning [11], which build a network of states sampled at random from the configuration space. Alterovitz et al (2007) present a Stochastic Motion

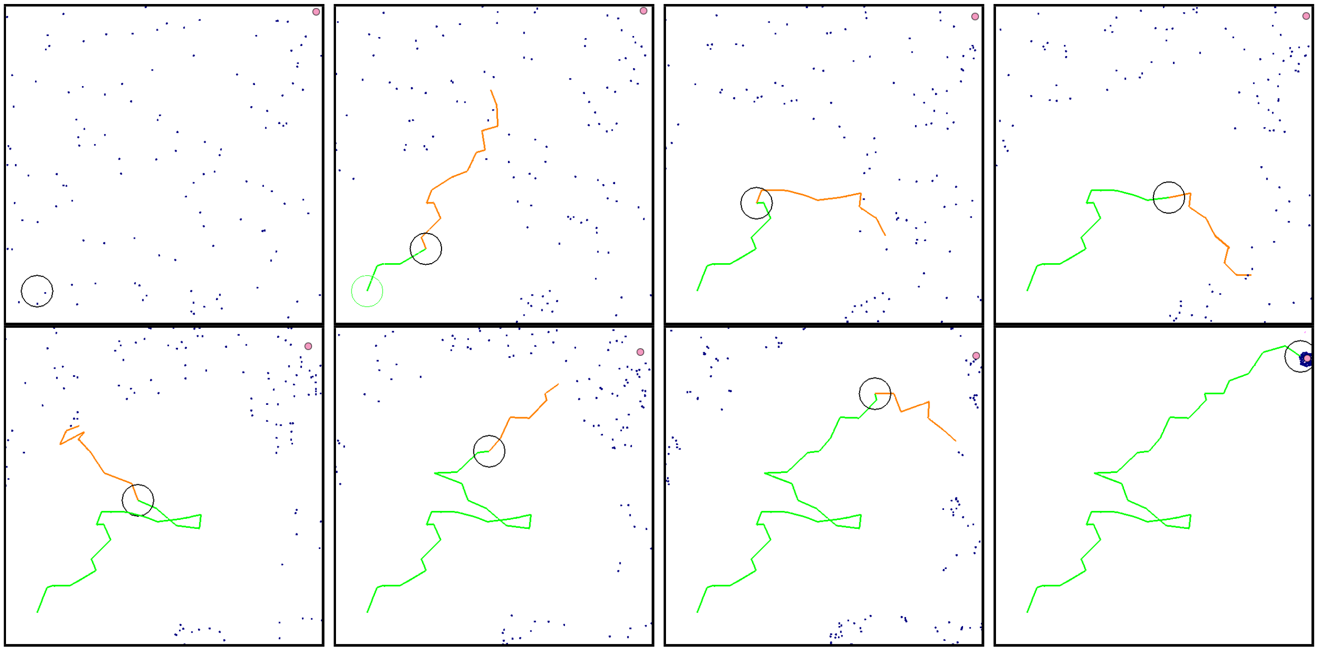

Fig. 1: An execution trace of a robot (large circle) searching for a wandering target (pink circle) in the unit square. The robot's sensor has a 0.25 unit range. The current belief state is represented by 100 particles (dots) and the current plan (orange) is updated by replanning.

Roadmap planner for continuous spaces with motion uncertainty, which solves an MDP using the discretization of state space induced by a PRM [1]. The techniques of Burns and Brock (2007) and Guibas et al (2008) augment roadmaps with edge costs for motions that have high probability of being in collision, and respectively address the problems of localization errors and environment sensing errors [3], [6]. Huang and Gupta (2009) address planning for manipulators under base uncertainty by associating probabilistic roadmaps with particles representing state hypotheses and searching for a short path that is likely to be collision free [10].

Another set of related approaches use assumptions of Gaussian observation and process noise, which makes planning much faster because probabilistic inference can be performed in closed form. The Belief Roadmap technique of Prentice and Roy (2009) computes a roadmap of belief states under both motion and sensing uncertainty, under the assumptions of Gaussian uncertainty and linear transition and observation functions [21]. van den Berg et al (2010) consider path planning while optimizing the likelihood that a path is collision-free, under the assumption that a Linear-Quadratic-Gaussian feedback controller is used to follow the path. Platt et al (2010) and du Toit and Burdick (2010) construct plans using a maximum-likelihood observation assumption, and correcting for observation errors by replanning [5], [19]. RBSR also uses a replanning strategy, but uses a particle-based uncertainty representation that is better at handling nonlinear and multimodal distributions, and makes no assumptions on the type of observations received.

Recently we discovered a similarity between RBSR and the "Rollout" approach of Bertsekas and Castañon (1999) presented in the context of a sequential stochastic decision problem [2]. The major difference is that RBSR uses a Monte-Carlo search, biased by a Voronoi distance heuristic, to generate a much deeper rollout search tree. Our analysis applies to much more general POMDPs as well.

Many authors have used *macro-actions* to improve the efficiency of planning in MDPs and POMDPs by reducing exploration breadth and depth [16]. Hsiao et. al. (2008) addressed a robot grasping problem using specially constructed macro-actions that either provide information or seek the goal [9]. They demonstrate that if uncertainty grows slowly during information-gathering, then forward planning can be limited to depth one. RBSR can also be interpreted as depth-one forward planning, using the QMDP policy as a goal-seeking macro-action and belief-space sampling to produce information-gathering macro-actions on the fly. Two other recent works have also tackled the problem of constructing macro-actions automatically and with increasing granularity during forward planning [8], [13]. These approaches are limited to macro-actions that reach subgoal states, and we suspect that RBSR constructs better information-gathering macro-actions using belief space criteria; on the other hand we also suspect that the approaches in [8], [13] construct more optimal plans by searching to a greater observation depth.

## III. Problem Definition

RBSR is an "online" policy [22] that interleaves planning and execution steps much like a receding-horizon controller. Each iteration performs the following steps:

1) The planner searches among open-loop actions by building a tree rooted at the current belief state.
2) The planner evaluates the quality of the open-loop plans concatenated with a goal-seeking closed-loop policy.
3) The robot executes the action associated with the best branch out of the root node.
4) The sensor reading is observed, and the robot's belief state is updated using a particle filter.

This section describes the POMDP formulation and the space of policies over which it searches.

### A. POMDP Modeling

The problem is formalized as an undiscounted partially-observable Markov decision process (POMDP) over a set of states $S$, actions $A$, and observations $O$. In our analysis we will treat $S$, $A$, and $O$ as discrete sets, but all of the components of the analysis and algorithm can be extended to continuous sets in a straightforward manner. A *belief state* $b(s)$ is defined to be a probability distribution over $S$. The robot starts at an initial belief state $b_{init}$ and wishes to maximize expected reward. There is a goal set $G \subseteq S$ terminal states. At discrete time steps the robot performs an action, which changes its (unobserved) state, and it receives an observation.

The dynamics of the system are specified in the *transition model* $\mathcal{T} : s, a, s' \rightarrow Pr(s'|s, a)$ that gives the probability of arriving in state $s'$ when executing action $a$ at state $s$. The *sensor model* $\mathcal{O} : s, o \rightarrow Pr(o|s)$ specifies the probability of observing $o$ in state $s$.

We extend the notation $\mathcal{T}$ to define the operator that returns the successor belief state after executing action $a$:

$$\mathcal{T}(b; a)(s) = \int_{s' \in S} \mathcal{T}(s, a, s')b(s')ds' \tag{1}$$

And the posterior belief state after observing $o$:

$$\mathcal{T}(b; a; o)(s) = \frac{1}{Z} \int_{s' \in S} \mathcal{T}(s, a, s')\mathcal{O}(s, o)b(s')ds' \tag{2}$$

where $Z$ is a normalization factor that ensures that the integral of the distribution is equal to 1. Furthermore, we use the notation $\mathcal{T}(b; a_1, \ldots, a_n; o_1, \ldots, o_n)$ to describe the resulting belief state after $n$ actions and observations.

Given a reward function $R(s, a)$ and discount factor $\gamma$ (we do allow undicounted $\gamma$=1), the quality of a belief-space policy $\pi : b \rightarrow a$ is measured by the expected return

$$V_\pi(b) = E_{s \sim b}[R(s, \pi(b)) + \gamma \int_{o \in O} V_\pi(\mathcal{T}(b; \pi(b); o))dPr(o|s)]. \tag{3}$$

### B. Open-Loop Information Gathering Online Planning

RBSR makes use open-loop information gathering in conjunction with a goal-seeking policy $\pi_G$ that performs well when uncertainty is low. It searches explicitly for plans that help $\pi_G$ attain the goal, which implicitly generates information-gathering actions as long as the observations attained along the open-loop plan are useful.

We define the "ideal" $\pi_{RBSR}$ as computing the open-loop plan $a^{(n)} = \{a_1, \ldots, a_n\}$ that optimizes the following objective function:

$$\max_{n>0, a^{(n)}} f(b; a^{(n)}, \pi_G) \tag{4}$$

where $f(b; \pi)$ is an *evaluation function* that is usually taken to be $V_\pi(b)$, the expected return of $\pi$, although we have considered an alternative definition as well. The notation $f(b; a^{(n)}, \pi_G)$ indicates that we concatenate the actions $a^{(n)}$ performed in open-loop fashion with the policy $\pi_G$ followed thereafter.

In practice, $b$, $f$, and the optimization must be approximated using sample-based techniques if $n$, $|S|$, or $|A|$ are large. We will set aside approximation issues until Section V, and for now we assume that all components are evaluated exactly.

### C. The QMDP Goal-Seeking Strategy

As a goal-seeking strategy $\pi_G$, RBSR uses the QMDP heuristic policy $\pi_{QMDP}$ that is quite successful in practice for highly-localized belief states or when information can be gathered quickly to localize the state (see Figure 2). Other strategies may also be used for $\pi_G$ as long as they satisfy a certain convexity requirement, as described below.

The QMDP policy essentially takes the optimal action assuming full observability is attained on the next step [16]. Suppose we are given a complete value function $V_{MDP}$ for the fully-observable version of the POMDP (computed by, e.g., value iteration). The belief-space policy $\pi_{\text{QMDP}}(b)$ is defined to ascend the expected value of $V_{MDP}$ over the distribution of states in $b$. More precisely, we define

$$V_{QMDP}(b) \equiv E_{s \sim b}[V_{MDP}(s)] = \int_{s \in S} V_{MDP}(s)b(s)ds \tag{5}$$

and define $\pi_{\text{QMDP}}$ to pick the action that ascends $V_{QMDP}(b)$ as quickly as possible:

$$\pi_{\text{QMDP}}(b) \equiv \arg\max_a V_{QMDP}(\mathcal{T}(b, a)). \tag{6}$$

For large $|A|$, we compute the $\arg\max$ in (6) by sampling. Also note that generally $V_{QMDP} \neq V_{\pi_{\text{QMDP}}}$ because $V_{QMDP}$ measures the optimal return under MDP dynamics while $V_{\pi_{\text{QMDP}}}$ measures the return of $\pi_{\text{QMDP}}$ under POMDP dynamics.

## IV. Analysis

RBSR will take an open-loop plan if the information gathered along the plan increases $f$ above its value at the initial state; otherwise it will follow QMDP. Hence, we must study the conditions under which $f$-increasing plans exist. We find that they exist unless process noise is high, or high initial uncertainty require multiple conditional information gathering actions (Figure 3). This suggests the need for measures of problem complexity based on rates of information gain or loss rather than coding size. We propose some preliminary quantities here.

RBSR will choose information gathering if it finds a sequence of actions $a^{(n)}$ such that $f(b; a^{(n)}, \pi_G) > f(b, \pi_G)$.

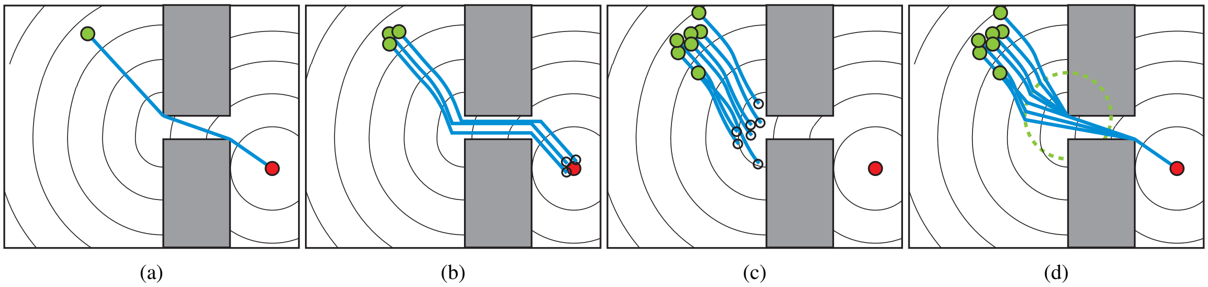

Fig. 2: The QMDP policy will succeed for well-localized belief states (a,b), but it may fall into local minima for a poorly localized belief state (c). On the other hand, QMDP allows the robot to incorporate new information during execution. So, if it could sense the corner of the obstacle as a landmark, then QMDP will also reach the goal (d).

The key step of our analysis is to consider the increase in success rate from new information gathered over the actions $a^{(n)}$

$$\alpha(b, a^{(n)}) = f(b; a^{(n)}, \pi_G) - \sum_{k=1}^{n} \gamma^{k-1} R(b_k, a_k) - \gamma^n f(\mathcal{T}(b; a^{(n)}); \pi_G) \tag{7}$$

as compared to the reduction in success rate due to the information lost in process noise:

$$\beta(b, a^{(n)}) = f(b; \pi_G) - \sum_{k=1}^{n} \gamma^{k-1} R(b_k, a_k) - \gamma^n f(\mathcal{T}(b; a^{(n)}); \pi_G). \tag{8}$$

So, RBSR will continue information gathering as long as there exist plans $a^{(n)}$ such that $\alpha(b, a^{(n)}) > \beta(b, a^{(n)})$. Broadly speaking, RBSR will work well in problems where $\alpha$ is large (useful information is accessible) and $\beta$ is small (process noise is low). We can further examine $\alpha$ and $\beta$ as follows.

**Non-negativity of $\alpha$.** First we define convexity as follows.

*Definition*: If

$$f(b; \pi) \leq p_1 f(b_1; \pi) + p_2 f(b_2; \pi) \tag{9}$$

whenever the belief state $b(s)$ can be decomposed according to $b(s) = p_1 b_1(s) + p_2 b_2(s)$ where $p_1$ and $p_2$ are nonnegative constants, then $\pi$ is said to be *convex* in $f$.

While it is well-known that the value function of a finite-horizon optimal policy is convex, convexity of suboptimal policies are not guaranteed. In particular it is possible to construct pathological cases in which $\pi_{QMDP}$ is not convex. Nevertheless we do not expect such cases to arise often in practice and $\pi_{QMDP}$ can be considered *de facto* convex. If this assumption is made then the following lemma can be reasonably assured to hold:

*Lemma*. If $\pi_G$ is convex in $f$, then $\alpha(b, a^{(n)})$ is nonnegative for all $b$ and $a^{(n)}$.

More specifically, this result holds as long as (9) holds over any disjoint partition of $O$ into subsets $O_1$, $O_2$, where $p_1$ and $p_2$ are the probabilities of the events $o \in O_1$ and $o \in O_2$ respectively with the state prior $b$, and $b_1$ and $b_2$ are the state distributions conditioned on $o \in O_1$ and $o \in O_2$ respectively.

**Boundedness of $\beta$.** Consider problems that obey a sort of Lipschitz continuity so that there exists a finite constant $M$ such that $\beta(b, a) \leq M$ for all $b$ and $a$. That is, the reduction of the evaluation function is bounded for any action. This implies that $\beta(b, a^{(n)}) \leq Mn$. This also implies that if there is a belief state $b$ that provides immediate information gain $x = \alpha(b, a)$ for some $a$, then an increase in $f$ is ensured for all belief states that can reach $b$ within $\lfloor x/M \rfloor$ actions if $\gamma = 1$, or $\lfloor \log((1-\gamma)x/M - 1)/\log(1/\gamma) \rfloor$ if $\gamma < 1$. Hence if the reachable belief space is "peaked" with informative belief states and $M$ is small, then RBSR will work well.

In robotics one frequently encounters problems in which uncertainty lies only in the initial belief on a constant or deterministically changing environmental parameter, like the mass of an object, a camera parameter to be calibrated, or an unknown map. In such cases the constant $M$ is identically zero and the chance of success is monotonically nondecreasing in the number of observations gathered (although it may be zero, as in Figure 3.b).

## V. RBSR Implementation in Continuous Spaces

The main benefit of RBSR is that it is applied relatively easily in continuous spaces. The original presentation of RBSR used the following scalable implementation [7]:

1) Belief states are represented as a weighted set of state hypotheses $\{(w_i, s_i)\}_{i=1}^{n}$ and the update in (2) is computed using particle filtering techniques [4].
2) For open-loop exploration RBSR uses a Voronoi exploration strategy to distribute $N$ belief states sparsely.
3) The MDP value function (5) is approximated by sampling the state space in precomputation.
4) The evaluation function $f$ is approximated using Monte-Carlo simulation of $m$ holdout particles.
5) An *expected information gain* score $EIG(b)$ is used to restrict the evaluations of $f$ to a small subset of nodes $M << N$. Because $EIG(b)$ is less expensive than $f$ to compute but is generally a good indicator of useful information, this strategy leads to major speed gains.

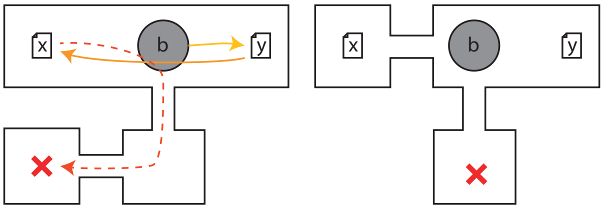

Fig. 3: (a) RBSR can find open-loop information gathering plans that integrate multiple sensor readings (solid line) as long as the resulting belief states lead to better goal-seeking policies (dotted line). (b) A case where open-loop exploration fails to make progress. A conditional plan is needed to reach the goal (bottom room) because the robot must first localize its $y$ coordinate (this information is located in the right room) in order to localize its $x$ coordinate (left room).

### A. Voronoi-Biased Exploration Strategy

The Voronoi-biasing exploration strategy is much like the Rapidly-Exploring Random Tree (RRT) motion planner [15] and is designed to cover the space of reachable open-loop motions quickly. To expand the tree, we sample a random target point $s_{tgt}$ from the state space $S$, and sample a set of representative particles from all belief states in the tree $R = \{s | b \in T, s \sim b\}$. Then, we find the closest point $s$ from $R$ to $s_{tgt}$. We then find a control $a$ action that brings $s$ closer to $s_{tgt}$. This procedure is repeated $N$ times.

### B. Value Function for the Fully-Observable MDP

In discrete POMDPs $V_{MDP}$ can be computed through a variety of methods, e.g., value iteration. In continuous POMDPs some method of discretization is needed. Following the manner of the Stochastic Motion Roadmap [1], we use randomly sampled set of states and actions and associate continuous states with the Voronoi cell of the sampled ones. This induces a discrete MDP that can be solved using value iteration. We then store each sampled state along with its $V_{MDP}$ value in a K-D tree for fast lookup.

### C. Monte-Carlo Estimation of Evaluation Function

To evaluate $f(b, \pi)$ we select a holdout set of $m$ particles $\{s^{(1)}, \dots s^{(m)}\}$ from $b$ which are used to simulate "ground truth". The complement of the holdout set $b'$ is used as the initial belief state. For each test sample $s^{(i)}$, $\pi$ is invoked from the initial belief state $b_0 = b'$, and $s_0 = s^{(i)}$ is used for simulating the "true" observations. The value of $f$ is taken to be the average sum of rewards.

In certain scenarios, particularly when the robot cannot achieve positive reward until after information gathering steps, we found that RBSR performance is greatly improved if $f$ is chosen not to evaluate expected return, but evaluate the sum of expected rewards until QMDP reaches a local minimum of $V(b)$, and then adding the value of $V(b)$. This encourages RBSR to gather information even if its utility is delayed.

### D. Expected Information Gain Scoring Strategy

An expected information gain strategy avoids running expensive evaluations of $f$ on belief states that are unlikely to yield improvements in $f$. The intuition is that information gain is a sort of proxy score for QMDP favorability because it measures the spread of a belief state distribution, and QMDP tends to succeed more when states are localized. We compute the expected information gain for a belief state $b$ as follows. The information gain of the observation $o$ is the Kullback-Leibler divergence between the posterior distribution $b_o \equiv Pr(s|o, b)$ and the prior $b \equiv Pr(s|b)$:

$$I(b_o||b) = \int_{s \in S} Pr(s|o, b) \log \frac{Pr(s|o, b)}{Pr(s|b)}. \tag{10}$$

Given a particle representation of belief states $b_o$ and $b$, we replace the distribution $Pr(s|b)$ using a kernel density estimator with Gaussian kernels centered on the particles in $b$, and approximate the integral by the weighted sum over the particles $s^{(i)}$ in $b_o$.

The expected information gain is simply the expectation of (10) over $o$:

$$EIG(b) = \int_{o \in O} Pr(o|b) I(b_o||b) \tag{11}$$

We approximate this value using Monte Carlo integration by sampling an observation for each particle in $b$.

### E. Parameter Tuning

Each plan has time complexity $O(poly(n, N, m, M))$, where $poly$ is a low-degree polynomial with no term greater than degree 4, and space complexity $O((n + m)N)$. The parameters also affect likelihood of convergence. A low $N$ suffices in information-rich problems where a large fraction of open-loop plans yield useful information; otherwise $N$ must be large. $m$ affects the accuracy of the estimate of $f(b)$ because the standard error is proportional to $1/\sqrt{m}$. So, a low $m$ will yield more erratic information gathering plans.

The parameter $n$ affects how accurately RBSR tracks and predicts belief states using the particle filter, and should be set high enough to attain a desired accuracy. Note also that the problem designer must select from dozens of existing particle filtering techniques, each of which has different characteristics with respect to accuracy, computational complexity, and assumptions on problem structure [4]. In our experiments we do a small amount of tuning to find a parameter and particle filtering technique that yields reasonably accurate belief state tracking over long horizons.

## VI. Experimental Results

### A. Discrete POMDP Benchmarks

We evaluated RBSR against QMDP [16] and several modern offline, point-based POMDP solvers: PBVI [18], HSVI [23], HSVI2 [24], and SARSOP [14]. The purpose of this comparison is not to treat RBSR in competition to these techniques but rather to explore how the heuristic information gathering

| Problem | Planner | Return | Time (s) |
|---|---|---|---|
| Tiger-Grid | QMDP | 0.26 | 0.026 |
| (36s 5a 17o) | PBVI | 2.25 | 3448 |
| | HSVI | 2.35 | 10341 |
| | HSVI2 | 2.3 | 52 |
| | SARSOP | 2.35 | 52 |
| | RBSR | 2 | 0.44 |
| Hallway | QMDP | 0.14 | 0.012 |
| (61s 5a 21o) | PBVI | 0.53 | 288 |
| | HSVI | 0.52 | 10836 |
| | HSVI2 | 0.52 | 2.4 |
| | SARSOP | 0.53 | 7.8 |
| | RBSR | 0.46 | 0.46 |
| Hallway2 | QMDP | 0.052 | 0.02 |
| (93s 5a 17o) | PBVI | 0.34 | 360 |
| | HSVI | 0.35 | 10010 |
| | HSVI2 | 0.35 | 1.5 |
| | SARSOP | 0.34 | 10 |
| | RBSR | 0.31 | 0.46 |
| Tag | QMDP | -16.48 | 0.07 |
| (870s 5a 30o) | PBVI | -9.18 | 180880 |
| | HSVI | -6.37 | 10113 |
| | HSVI2 | -6.36 | 24 |
| | SARSOP | -6.13 | 6 |
| | RBSR | -9.51 | 0.87 |
| RockSample[4,4] | QMDP | 3.5 | 0.008 |
| (257s 9a 2o) | PBVI | 17.1 | 2̃000 |
| | HSVI | 18 | 511 |
| | HSVI2 | 18 | 0.75 |
| | SARSOP | 18.1 | 0.52 |
| | RBSR | 9.76 | 0.16 |
| RockSample[7,8] | QMDP | 0 | 2.1 |
| (12545s 13a 2o) | HSVI | 15.1 | 10266 |
| | HSVI2 | 20.6 | 1003 |
| | SARSOP | 21.3 | 400 |
| | RBSR | 13 | 4.29 |

TABLE I: Comparing RBSR to state-of-the-art approximate planners on discrete planning benchmarks. For RBSR, the Time column indicates time per iteration.

strategy of RBSR performs relative to optimal policies, and how well it scales to larger domains.

RBSR was run for 1000 trials on a single thread of a 2.67GHz Intel Core i7 PC, using exact belief state representations and parameter settings $N = 1000$, $M = 100$, and $m = 100$. Table I shows expected return and running time for several discrete planning benchmarks. The number of states, actions, and observations in each problem are listed. Values for PBVI, HSVI, HSVI2, and SARSOP were taken from [14], [23] and are run on comparable or slightly faster machines than our test machine. RBSR attains drastically better rewards than QMDP and, surprisingly, often approaches the reward level of PBVI. It performs significantly worse than the newer HSVI and SARSOP planners on the Tag and RockSample problems. We believe this is the case because these problems require weighing multiple sensor readings and conditional plans into an optimal decision, and so the open-loop information gathering plans of RBSR are suboptimal.

### B. Continuous Pursuit Scenario

Our first continuous experiment is a 2D pursuit scenario with a 4D state space. The robot must intercept a slower target that moves at random inside a unit square (Figure 1). The position of the robot is observable and controlled precisely, but it cannot sense the target outside a circle of radius 0.25. The target's position is a uniform distribution in the initial belief state, and the goal condition is to achieve a distance of 0.05 to the target. We tested three sensor models: 1) a position sensor that reports the target's $x, y$ position relative to the robot, 2) a direction sensor that reports only direction and not distance, and 3) a distance sensor that does not report direction.

Although this is not difficult to solve using special-purpose strategies, it poses a challenge for general-purpose planners to solve in a reasonable amount of time and memory. For example, the SARSOP planner [14] can approximately solve a coarsely discretized version of the problem in a few minutes, but it exhausts our test machine's 2Gb of memory once the resolution of the workspace grid exceeds 15x15.

Using preliminary experiments we tuned the number of particles in the belief state needed for accurate particle filtering, and found that 100 particles were sufficient for the position and direction sensor, and 200 particles were needed for the proximity sensor. So, we used $m = 50$ particles as a holdout set, and $n = 150$ and $n = 250$, respectively, for the position/direction sensors and the proximity sensor. In 25 trials on each of these problems, with random target start states, RBSR never failed to reach the target. Several execution traces for different initial target positions are drawn in Figure 4. Average path length is approximately 1.7, which is close to the expected path length computed by SARSOP on a 15 x 15 grid, but is still suboptimal. Each replanning iteration took about 15 s on average, with standard deviation $\sim$ 10 s.

### C. Continuous Localization Scenario

Our second continuous scenario places a robot at an unknown configuration in a known $d$-dimensional environment, in which it must localize itself and reach a small goal by measuring the distance to obstacles. The sensor has a limited range, which requires that the robot perform several steps of active sensing before reaching the goal. The optimal strategy is to proceed toward a wall until the sensor returns a reading, and then proceed to an adjacent wall until a closer reading is obtained, and so on until it achieves $d$ readings from $d$ linearly independent walls. Note that RBSR does not have a "proceed until" action in its action set, so instead it must approximate such a policy by a sequence of conditional movement actions and sensing actions.

Figure 5 depicts a solution trace where $d = 2$, $\mathcal{S}$ is the unit square, the initial belief state is a circular Gaussian distribution with standard deviation 0.1, and the goal radius and the sensing radius are both set to 0.05. To represent belief states we used 150 particles with a holdout set of 50. We also tested a space containing obstacles (Figure 6). In both examples, RBSR performs localization by moving close to obstacle boundaries, in somewhat random fashion, until it senses nearby walls. This continues until sufficient data is gathered to reach the goal.

We also tested scalability with respect to dimension. Figure 7 plots the convergence of taken by RBSR in problems from $d = 5$ to $d = 8$ in the unit hypercube. The number of

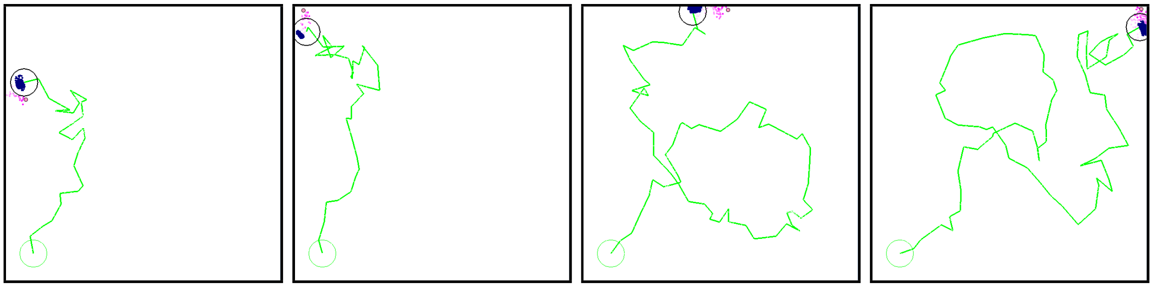

Fig. 4: Execution traces of the pursuit example for four different initial target locations (purple circles). The robot uses a distance sensor with maximum range 0.25.

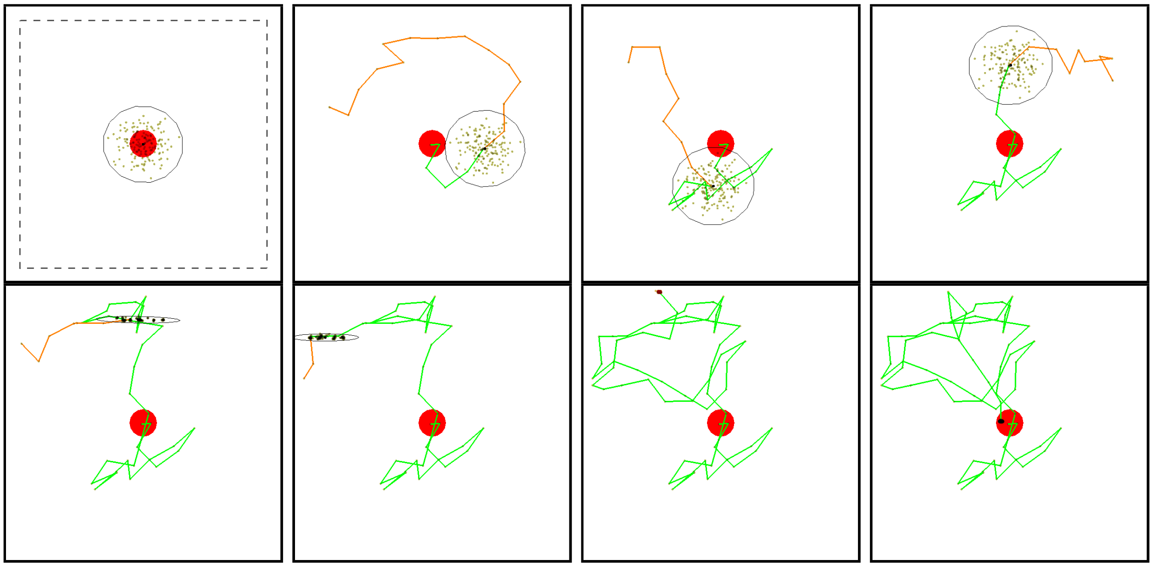

Fig. 5: An execution trace of a robot localizing itself to reach the red circle with high probability. Its sensor measures the distance to the walls, and has maximum range 0.05 (dashed lines). The current belief state is represented by 100 particles (dots) with a covariance ellipsoid, and the current plan (orange) is updated by replanning.

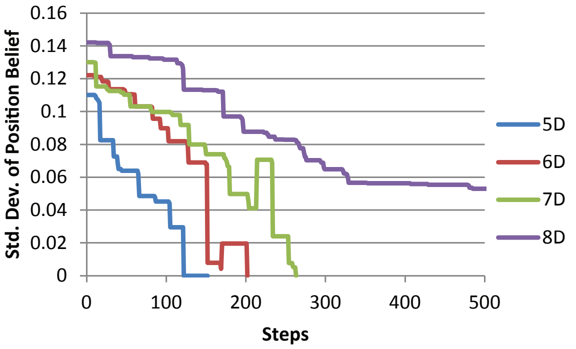

Fig. 7: Traces of the belief state standard deviation of boundary localization problems in unit hypercubes ranging from 5-D to 8-D. The 8D problem fails to converge after 1000 iterations. (Spikes are due to particle filtering artifacts)

particles is set to $n = 500$, but we kept all other parameters unchanged from the experiment in Figure 5. Running time per timestep is roughly linear in dimension, ranging from approximately 6 s in the 3D case up to approximately 28 s in the 8D case. We found that convergence dropped off sharply: in up to 7 dimensions, RBSR solved 10/10 trials, while in 8 dimensions, it solved none. We suspect that this is due to the increased distance and sparsity of states in which sensor readings are informative, which requires deeper exploration and more carefully selected information gathering plans.

## VII. Conclusion

This paper analyzed a Randomized Belief-Space Replanning (RBSR) technique for partially-observable problems in continuous state spaces. It constructs partial plans by sampling open-loop actions at random, and by evaluating the quality of future belief states by simulating a QMDP-like policy that performs well when the state is well-localized. By iteratively incorporating sensor feedback from plan execution and replanning, RBSR avoids having to compute a policy over large belief spaces. Experiments show that it solves a target pursuit problem with a 4D state space and a localization problem in up to 7D.

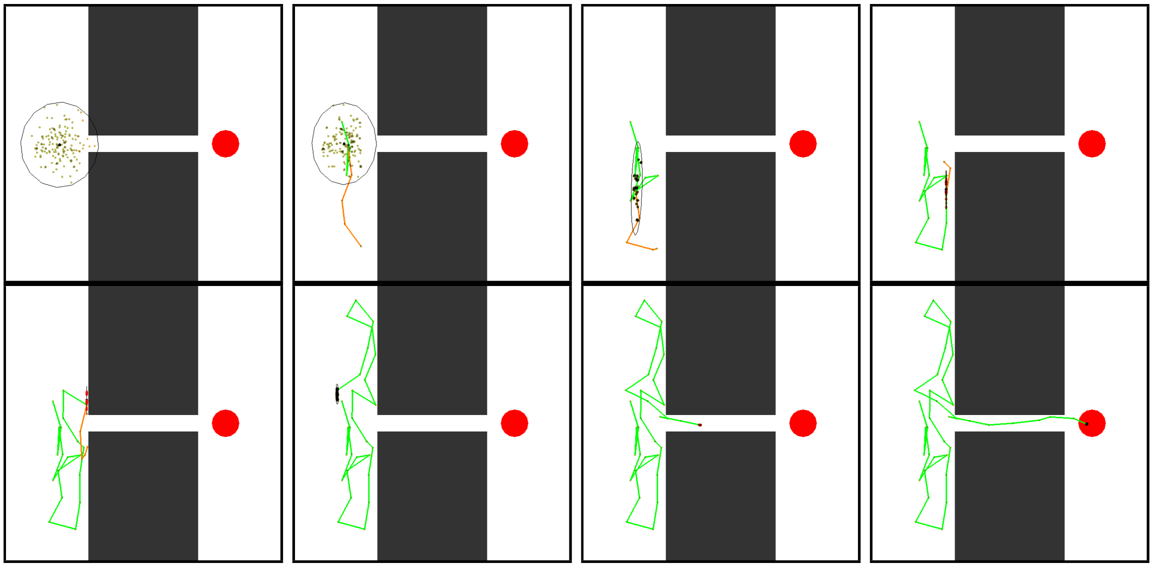

Fig. 6: A robot localizing itself using a proximity sensor in a space with obstacles.

Although we have identified certain problem characteristics related to information gain and loss rates that govern RBSR's convergence, it is unclear whether these characteristics are computable over realistic problem classes. RBSR also produces inefficient plans with jerky artifacts due to randomized exploration, and perhaps path smoothing will lead to more efficient exploration. Future benchmarks for continuous POMDPs will also help with rigorous comparison of planner performance.

## References

[1] R. Alterovitz, T. Simeon, and K. Goldberg. The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty. In *Robotics: Science and Systems*, June 2007.

[2] D. P. Bertsekas and D. A. C. non. Rollout algorithms for stochastic scheduling problems. *Journal of Heuristics*, pages 89–108, 1999.

[3] B. Burns and O. Brock. Sampling-based motion planning with sensing uncertainty. In *Proc. IEEE Int. Conf. on Robotics and Automation*, 2007.

[4] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.

[5] N. du Toit and J. Burdick. Robotic motion planning in dynamic, cluttered, uncertain environments. In *IEEE Int. Conf. on Robotics and Automation*, 2010.

[6] L. J. Guibas, D. Hsu, H. Kurniawati, and E. Rehman. Bounded uncertainty roadmaps for path planning. In *Workshop on the Algorithmic Foundations of Robotics*, Guanajuato, Mexico, 2008.

[7] K. Hauser. Randomized belief-space replanning in partially-observable continuous spaces. In *Workshop on the Algorithmic Foundations of Robotics*, 2010.

[8] R. He, E. Brunskill, and N. Roy. Puma: Planning under uncertainty with macro-actions. In *Proc. Twenty-Fourth Conf. on Artificial Intelligence (AAAI)*, 2010.

[9] K. Hsiao, T. Lozano-Perez, and L. P. Kaelbling. Robust belief-based execution of manipulation programs. In *Eighth International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2008.

[10] Y. Huang and K. Gupta. Collision-probability constrained prm for a manipulator with base pose uncertainty. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1426–1432, Piscataway, NJ, USA, 2009. IEEE Press.

[11] L. E. Kavraki, P. Svetska, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. and Autom.*, 12(4):566–580, 1996.

[12] M. Kearns, Y. Mansour, and A. Y. Ng. Approximate planning in large pomdps via reusable trajectories. In *Advances in Neural Information Processing Systems 12*. MIT Press, 2000.

[13] H. Kurniawati, Y. Du, D. Hsu, and W. Lee. Motion planning under uncertainty for robotic tasks with long time horizons. In *Proc. Int. Symp. on Robotics Research*, 2009.

[14] H. Kurniawati, D. Hsu, , and W. Lee. Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In *Proc. Robotics: Science and Systems*, 2008.

[15] S. M. LaValle and J. J. Kuffner, Jr. Rapidly-exploring random trees: progress and prospects. In *WAFR*, 2000.

[16] M. Littman, A. R. Cassandra, and L. P. Kaelbling. Learning policies for partially observable environments: Scaling up. In *Proc. 12th Int. Conf. on Machine Learning*, pages 362–370. Morgan Kaufmann, 1995.

[17] M. L. Littman, J. Goldsmith, and M. Mundhenk. The computational complexity of probabilistic planning. In *Journal of Artificial Intelligence Research*, volume 9, pages 1–36, 1998.

[18] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1025–1032, Acapulco, Mexico, Aug 2003.

[19] R. Platt, R. Tedrake, L. Kaelbling, and T. Lozano-Perez. Belief space planning assuming maximum likelihood observations. In *Proc. Robotics: Science and Systems*, 2010.

[20] J. M. Porta, N. Vlassis, M. T. J. Spaan, and P. Poupart. Point-based value iteration for continuous pomdps. *Journal of Machine Learning Research*, 7:2329–2367, 2006.

[21] S. Prentice and N. Roy. The Belief Roadmap: Efficient Planning in Belief Space by Factoring the Covariance. *The International Journal of Robotics Research*, 28(11-12):1448–1465, 2009.

[22] S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. Online planning algorithms for pomdps. *Journal of Artificial Intelligence Research*, 32:663–704, 2008.

[23] T. Smith and R. Simmons. Heuristic search value iteration for pomdps. In *Proc. Uncertainty in Artificial Intelligence*, page 520527, 2004.

[24] T. Smith and R. Simmons. Point-based pomdp algorithms: Improved analysis and implementation. In *Proc. Uncertainty in Artificial Intelligence*, 2005.

[25] S. Thrun. Monte carlo pomdps. In *Advances in Neural Information Processing Systems 12 (NIPS-1999)*, page 10641070. MIT Press, 2000.