Point-Based Policy Synthesis for POMDPs With Boolean and Quantitative Objectives

Yue Wang, Swarat Chaudhuri, and Lydia E. Kavraki

***Abstract*—Effectively planning robust executions under uncertainty is critical for building autonomous robots. Partially observable Markov decision processes (POMDPs) provide a standard framework for modeling many robot applications under uncertainty. We study POMDPs with two kinds of objectives: (1) Boolean objectives for a correctness guarantee of accomplishing tasks and (2) quantitative objectives for optimal behaviors. For robotic domains that require both correctness and optimality, POMDPs with Boolean and quantitative objectives are natural formulations. We present a practical policy synthesis approach for POMDPs with Boolean and quantitative objectives by combining policy iteration and policy synthesis for POMDPs with only Boolean objectives. To improve efficiency, our approach produces *approximate* policies by performing the point-based backup on a small set of representative beliefs. Despite being approximate, our approach maintains validity (satisfying Boolean objectives) and guarantees improved policies at each iteration before termination. Moreover, the error due to approximation is bounded. We evaluate our approach in several robotic domains. The results show that our approach produces good approximate policies that guarantee task completion.**

***Index Terms*—Formal Methods in Robotics and Automation, Task Planning, Motion and Path Planning.**

I. Introduction

DEPLOYING robots in the physical world presents a fundamental challenge with planning robust executions under uncertainty. The framework of POMDPs [1] offers a standard approach for modeling robot tasks under uncertainty.

A key algorithmic problem for POMDPs is the synthesis of *policies* [1] that specify actions to take for all possible events during execution. Traditionally, policy synthesis for POMDPs focuses on quantitative objectives such as (discounted) rewards [2]–[11]. Recently, there has been a growing interest in POMDPs with *boolean* objectives [12]–[17] that require accomplishing tasks under all possible events.

On one hand, POMDPs with quantitative objectives provide an optimality guarantee but may lead to overly conservative or overly risky behaviors [18], depending on the reward function chosen. On the other hand, POMDPs with boolean objectives provide a strong correctness guarantee of completing tasks [12]–[17] but the constructed policy may not be optimal. For the example domain shown in Fig. 1, there are many valid policies that can accomplish the task, i.e., satisfying the boolean objective. Among these valid policies, the most preferable policy is the one that passes the smallest number of red regions that the robot should avoid. Therefore, for domains that require both correctness and optimality, POMDPs with boolean and quantitative objectives are natural formulations.

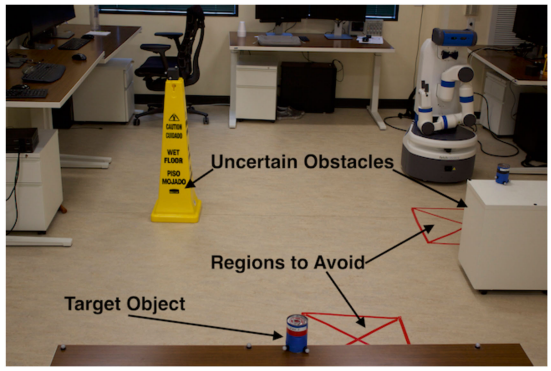

Fig. 1. A robot with imperfect actuation and perception needs to pick up the blue can on the table while avoiding collisions with uncertain obstacles such as floor signs and file cabinets (Boolean objective). There are two regions marked with red tape that the robot should avoid (quantitative objective).

Policy synthesis for POMDPs with boolean and quantitative objectives has been studied before [15]. In that work, the goal is to find an optimal policy that also ensures a goal state is reached with *probability 1*. A more general policy synthesis problem of POMDPs with both boolean and quantitative objectives is to synthesize an optimal policy that satisfies the boolean objective with *a probability above a threshold*. In this letter, we study this problem for the special case of *safe-reachability* objectives, which require that with a probability above a threshold, a goal state is eventually reached while keeping the probability of visiting unsafe states below a different threshold. Many robot tasks such as the one shown in Fig. 1 can be formulated as POMDPs with safe-reachability and quantitative objectives.

POMDPs with only safe-reachability objectives have been considered in previous works [16], [17]. An offline approach called Bounded Policy Synthesis (BPS) is presented in [16]. BPS computes a valid policy over a *goal-constrained belief space* rather than the entire belief space to improve scalability. The goal-constrained belief space only contains beliefs visited by desired executions that can achieve the safe-reachability objective and is generally much smaller than the original belief space. For POMDPs with only quantitative objectives, point-based POMDP solvers [3]–[10] have been quite successful in recent years. Point-based POMDP solvers can solve large POMDPs by producing approximate policies over representative beliefs instead of the entire belief space.

Manuscript received September 10, 2018; accepted January 21, 2019. Date of publication February 7, 2019; date of current version February 28, 2019. This letter was recommended for publication by Associate Editor J. O'Kane and Editor D. Song upon evaluation of the reviewers' comments. This work was supported by Grants NSF CCF 1139011, NSF CCF 1514372, NSF CCF 1162076, and NSF IIS 1317849. *(Corresponding author: Yue Wang.)*

The authors are with the Department of Computer Science, Rice University, Houston, TX 77005 USA (e-mail: yw27@rice.edu; swarat@rice.edu; kavraki@rice.edu).

Digital Object Identifier 10.1109/LRA.2019.2898045

Ideally, we can construct *exact* policies for POMDPs with safe-reachability and quantitative objectives by enumerating all beliefs in the goal-constrained belief space and performing value iteration on these beliefs. However, this enumeration is generally expensive [19] even though the goal-constrained belief space with a bounded horizon is finite. We take inspiration from recent advances in point-based POMDP solvers to improve efficiency by selecting representative beliefs from the goal-constrained belief space and producing *approximate* policies through point-based backup [3], [5] over these representative beliefs rather than the entire goal-constrained belief space. For previous point-based POMDP methods, this selection of representative beliefs is typically done through sampling from reachable belief space. In our case, sampling from the reachable belief space may not work well due to the additional boolean objectives: the sampled belief may not be in the goal-constrained belief space and thus there are no valid policies from the sampled belief. The key problem here is the selection of representative beliefs from the goal-constrained belief space, which essentially asks whether there exists a valid policy starting from a belief. This selection process can be done by invoking our previous BPS method [16] to compute valid policies. Since we need to construct policies for approximating the goal-constrained belief space, we choose policy iteration to handle the quantitative objective because policy iteration typically converges faster than value iteration [4].

In this work, we present an offline policy synthesis approach, *Point-Based Policy Synthesis* (PBPS), for POMDPs with safe-reachability and quantitative objectives. PBPS combines BPS [16] and Point-Based Policy Iteration (PBPI) [4] to synthesize good approximate policies that satisfy the safe-reachability objective. At a high level, PBPS applies BPS to efficiently explore the goal-constrained belief space for finding a valid policy π that satisfies the safe-reachability objective. Then PBPS adapts PBPI to transform π into an improved policy π'. This improved policy π' may reach some belief b' that is not visited by the current policy. Therefore, PBPS invokes BPS again to check whether there exists a valid policy starting from b'. By doing this, we can explore new belief regions and expand the set of representative beliefs, which is crucial to the quality of the constructed policy [3]–[5]. PBPS alternates between the computation of valid policies and policy iteration until the termination condition is satisfied.

We prove that PBPS inherits many desirable properties of PBPI. First, PBPS maintains validity and is monotonic: at each iteration before termination, PBPS produces a valid policy with improved quality. Second, the error introduced by PBPS due to approximation is bounded. We evaluate PBPS in the kitchen domain [16] and the Tag domain [3]. We also validate PBPS on a Fetch robot for the domain in Fig. 1. The results demonstrate that PBPS produces good approximate policies that achieve the given safe-reachability objective.

Related Work: POMDPs [1] offer a principled mathematical framework for modeling many robotics applications under uncertainty. Many POMDP solvers [2]–[11] focus on quantitative objectives such as (discounted) rewards. Recently, there has been a large body of work in constrained POMDPs [18], [20]–[22], chance-constrained POMDP [23], and risk-sensitive POMDPs [24], [25] that handle explicit cost/risk constraints. There are two differences between these POMDPs and POMDPs with boolean and quantitative objectives. First, the objective of these POMDPs is to maximize the expected reward while keeping the *expected* cost/risk below a threshold, while our objective is to maximize the expected reward while satisfying a boolean objective *in every execution* including the worst case. Second, these POMDPs typically assign positive rewards for goal states to ensure reachability, while our boolean objectives directly encode reachability constraints. It has been shown that for certain domains that demand a correctness guarantee of accomplishing tasks, POMDPs with boolean objectives offer a better guarantee than these POMDPs [16].

Recent work [12]–[17] has studied POMDPs with boolean objectives. POMDPs with both boolean and quantitative objectives were first introduced in [15]. In that work, the authors studied the almost-sure satisfaction problem where the goal is to find an optimal policy that ensures a goal state is reached with probability 1. In this work, we focus on a more general policy synthesis problem of POMDPs with safe-reachability and quantitative objectives, where the goal is to find an optimal policy that ensures a goal state is reached with a probability above a threshold, while keeping the probability of visiting unsafe states below a different threshold.

Our method computes an approximate policy over a representative subset of the goal-constrained belief space based on the ideas from Point-Based Policy Iteration (PBPI) [4] and the previous method Bounded Policy Synthesis (BPS) [16] for POMDPs with only safe-reachability objectives. We apply the same BPS algorithm in [16] to efficiently explore the goal-constrained belief space and construct a valid policy. However, we can not directly apply PBPI to find an optimal policy for two reasons. First, PBPI considers infinite-horizon policies represented as finite-state controllers [26] while our approach focuses on bounded-horizon policies represented as a set of conditional plans [9]. Second, PBPI only considers quantitative objectives while our approach needs to consider both boolean and quantitative objectives. More details on how we adapt PBPI are discussed in Section III-A.

II. Problem Formulation

In this work, we consider the problem of policy synthesis for POMDPs with safe-reachability and quantitative objectives. We follow the notation in [16] and [17].

Definition 1 (POMDP): A POMDP is a tuple $P = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{Z}, r)$, where $\mathcal{S}$ is a *finite* set of states, $\mathcal{A}$ is a *finite* set of actions, $\mathcal{T}$ is a probabilistic transition function, $\mathcal{O}$ is a *finite* set of observations, $\mathcal{Z}$ is a probabilistic observation function, and r is a reward function. $\mathcal{T}(s, a, s') = \Pr(s'|s, a)$ specifies the probability of moving to state $s' \in \mathcal{S}$ after taking action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$. $\mathcal{Z}(s', a, o) = \Pr(o|s', a)$ specifies the probability of observing observation $o \in \mathcal{O}$ after taking action $a \in \mathcal{A}$ and reaching state $s' \in \mathcal{S}$. $r(s, a)$ defines the reward of executing action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$.

Due to uncertainty, statesare partially observable and typically we maintain a probability distribution (*belief*) over all states $b : \mathcal{S} \mapsto [0, 1]$ with $\sum_{s\in\mathcal{S}} b(s) = 1$. The set of beliefs $\mathcal{B} = \{b | \sum_{s\in\mathcal{S}} b(s) = 1\}$ is the *belief space*. The belief space transition $\mathcal{T}_\mathcal{B}$ is *deterministic*. $b_a^o = \mathcal{T}_\mathcal{B}(b, a, o)$ is the successor belief for a belief b after taking an action a and receiving an observation o, defined by Bayes rule: $\forall s' \in \mathcal{S}, b_a^o(s') = \frac{\mathcal{Z}(s',a,o)\sum_{s\in\mathcal{S}} \mathcal{T}(s,a,s')b(s)}{\Pr(o|b,a)}$, where $\Pr(o|b, a) = \sum_{s'\in\mathcal{S}} \mathcal{Z}(s', a, o) \sum_{s\in\mathcal{S}} \mathcal{T}(s, a, s')b(s)$ is the probability of

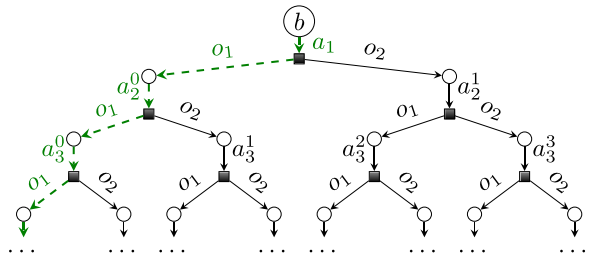

Fig. 2. A conditional plan. Circle nodes represent beliefs, edges from circle to rectangle nodes ($a_1, a_2^0, \dots 3$) represent actions, and edges from rectangle to circle nodes (o_1 and o_2) represent observations.

receiving an observation o after taking an action a in a belief b.

Definition 2 (k-Step Plan): A k-step *plan* is a sequence $\sigma = (b_0, a_1, o_1, \dots, a_k, o_k, b_k)$ such that for all $i \in (0, k]$, the belief updates satisfy the transition function $\mathcal{T}_B$, i.e., $b_i = \mathcal{T}_B(b_{i-1}, a_i, o_i)$, where $a_i \in \mathcal{A}$ is an action and $o_i \in \mathcal{O}$ is an observation. $|\sigma| = k$ is the length of the k-step plan σ.

Definition 3 (Safe-Reachability): A *safe-reachability objective* is a tuple $\mathcal{G} = (Dest, Safe)$, where $Safe = \{b \in \mathcal{B} | \sum_{s \text{ violates safety}} b(s) < \delta_2\}$ is a set of safe beliefs and $Dest = \{b \in Safe | \sum_{s \text{ is a goal state}} b(s) > 1 - \delta_1\} \subseteq Safe$ is a set of goal beliefs. δ_1 and δ_2 are small values for tolerance.

$\mathcal{G}$ compactly represents the set $\Omega_\mathcal{G}$ of valid plans:

Definition 4 (Valid k-Step Plan): A k-step plan $\sigma = (b_0, \dots, a_k, o_k, b_k)$ is *valid* w.r.t. a safe-reachability objective $\mathcal{G} = (Dest, Safe)$ if $b_k \in Dest$ is a goal belief and all beliefs visited before step k are safe beliefs ($\forall i \in [0, k), b_i \in Safe$).

Policy and Conditional Plan: Computing exact policies over the entire belief space $\mathcal{B}$ is intractable, due to the curse of dimensionality [27]: $\mathcal{B}$ is a high-dimensional space with an infinite number of beliefs. To make the problem tractable, we can exploit the *reachable belief space*$\mathcal{B}_{b_0}$ [3], [5]. $\mathcal{B}_{b_0}$ only contains beliefs reachable from the initial belief b_0 and is much smaller than $\mathcal{B}$.

Our previous BPS work [16] has shown that the performance of policy synthesis for POMDPs with safe-reachability objectives can be further improved based on the notion of a *goal-constrained belief space* $\mathcal{B}_\mathcal{G}$. $\mathcal{B}_\mathcal{G}$ combines the reachable belief space $\mathcal{B}_{b_0}$ and the set $\Omega_\mathcal{G}$ of valid plans defined by the safe-reachability objective $\mathcal{G}$. $\mathcal{B}_\mathcal{G}$ only contains beliefs reachable from the initial belief b_0 under a valid plan $\sigma \in \Omega_\mathcal{G}$ and is generally much smaller than the reachable belief space.

Previous results [28]–[30] have shown that the problem of policy synthesis for POMDPs is generally undecidable. However, when restricted to a bounded horizon, this problem becomes PSPACE-complete [27], [31]. Therefore, BPS computes a bounded policy π over the goal-constrained belief space $\mathcal{B}_\mathcal{G}$ within a bounded horizon h. This bounded policy π is essentially a set of conditional plans [9]:

Definition 5 (Conditional Plan): A conditional plan is a tuple $\gamma = (a, \nu)$, where a is an action and ν is an *observation strategy* that maps an observation o to a conditional plan γ'.

Fig. 2 shows an example of a conditional plan $\gamma = (a_1, \nu)$ represented as a tree rooted at a belief b. γ together with the belief b defines a set $\Omega_{\gamma,b}$ of plans in the belief space. For each plan $\sigma \in \Omega_{\gamma,b}$, the execution is the following process: initially, we start at the belief b and take the action a_1 specified by γ. Upon receiving an observation o, we move to the successor belief $b_a^o = \mathcal{T}_\mathcal{B}(b, a_1, o)$ and start executing the conditional plan γ_o for the observation o specified by the observation strategy ν. This process repeats until we reach a terminal belief. The horizon $h_\gamma = \max_{\sigma \in \Omega_{\gamma,b}} |\sigma|$ of a conditional plan γ is defined as the maximum length of the plans in $\Omega_{\gamma,b}$.

Definition 6 (Valid Conditional Plan): A conditional plan γ is *valid* starting from a belief $b \in \mathcal{B}$ w.r.t. a safe-reachability objective $\mathcal{G}$ if every plan in $\Omega_{\gamma,b}$ is valid ($\Omega_{\gamma,b} \subseteq \Omega_\mathcal{G}$).

Definition 7 (k-Step Policy): A k-step policy $\pi = \{\gamma_1, \gamma_2, \dots\}$ is a set of conditional plans.

A k-step π defines a set of *representative beliefs* $\mathcal{B}_\pi \subseteq \mathcal{B}$ that contains all beliefs visited by π. For every belief $b \in \mathcal{B}_\pi$, $\pi(b) = \gamma \in \pi$ is the conditional plan associated with this belief b. For a k-step policy π, the horizon $h_\pi = \max_{\gamma \in \pi} h_\gamma$ is k.

Similarly, the k-step policy $\pi = \{\gamma_1, \gamma_2, \dots\}$ defines a set $\Omega_\pi = \bigcup_{b \in \mathcal{B}_\pi} \Omega_{\gamma,b}$ of plans, where γ is the conditional plan specified for the belief b by the policy π.

Definition 8 (Valid k-Step Policy): A k-step policy π is *valid* w.r.t. a safe-reachability objective $\mathcal{G}$ if every plan in the set Ω_π is valid ($\Omega_\pi \subseteq \Omega_\mathcal{G}$).

Quantitative Objectives: Each conditional plan $\gamma = (a, \nu) \in \pi$ in a k-step policy $\pi = \{\gamma_1, \gamma_2, \dots\}$ induces a value function $V_\gamma(b)$ that specifies the expected total reward of executing the conditional plan γ starting from the belief b:

$$V_\gamma(b) = \sum_{s \in \mathcal{S}} r(s, a) b(s) + \sum_{o \in \mathcal{O}} \Pr(o|b, a) V_{\gamma_o}(b_a^o) \quad (1)$$

where $b_a^o = \mathcal{T}_\mathcal{B}(b, a, o)$ is the successor belief and γ_o is the conditional plan for the observation o specified by the observation strategy ν of the conditional plan γ.

Since the value function V_γ is linear with respect to the belief space [1], Eq. (1) can be rewritten as $V_\gamma(b) = \boldsymbol{\alpha}_\gamma \cdot b$, where $\boldsymbol{\alpha}_\gamma$ is the $\boldsymbol{\alpha}$-vector that specifies the reward for every state $s \in \mathcal{S}$ following the conditional plan $\gamma = (a, \nu)$:

$$\alpha_\gamma(s) = r(s, a) + \sum_{o \in \mathcal{O}} \sum_{s' \in \mathcal{S}} \mathcal{Z}(s', a, o) \mathcal{T}(s, a, s') \alpha_{\gamma_o}(s') \quad (2)$$

Therefore, the value function V_π of the policy π can be represented as a set of $\boldsymbol{\alpha}$-vectors $V_\pi = \{\boldsymbol{\alpha}_{\gamma_1}, \boldsymbol{\alpha}_{\gamma_2}, \dots\}$. For every belief $b \in \mathcal{B}_\pi$, $V_\pi(b) = V_\gamma(b) = \boldsymbol{\alpha}_\gamma \cdot b$, where γ is the conditional plan for the belief b specified by the policy π. For $b \notin \mathcal{B}_\pi$, $V_\pi(b) = \max_{\boldsymbol{\alpha} \in V_\pi} \boldsymbol{\alpha} \cdot b$.

Problem Statement: Given a POMDP P, an initial belief b_0, a safe-reachability objective $\mathcal{G}$, and a horizon bound h, our goal is to synthesize a *valid* k-step ($k \leq h$) policy $\pi_\mathcal{G}^* = \text{argmax}_{\text{valid}\pi} V_\pi(b_0)$ that maximizes the expected reward from b_0. Note that $\pi_\mathcal{G}^*$ is different from the optimal policy $\pi^* = \text{argmax}_\pi V_\pi(b_0)$ without the requirement of satisfying the safe-reachability objective (π^* may be invalid).

III. Point-Based Policy Synthesis

Fig. 3 shows the overview of our Point-Based Policy Synthesis (PBPS) approach (Algorithm 1). PBPS combines Bounded Policy Synthesis (BPS) [16] and Point-Based Policy Iteration (PBPI) [4] to compute a good approximate policy π that satisfies the given safe-reachability objective.

At a high level, PBPS aims to achieve both boolean (safe-reachability) and quantitative (maximizing rewards) objectives.

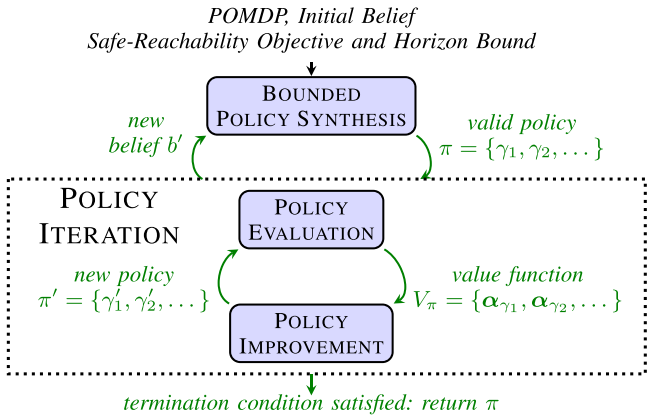

Fig. 3. Overview of the PBPS algorithm.

Algorithm 1: PBPS.

Input: POMDP $P = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{Z}, r)$, Initial Belief b_0, Safe-Reachability Objective $\mathcal{G}$, Horizon Bound h
Output: k-Step Policy π

1 $\pi \leftarrow \text{BPS}(P, b_0, \mathcal{G}, h)$ /* Initial valid policy */
2 **if** $\pi = \emptyset$ **then** /* No k-step ($k \leq h$) valid policy */
3 **return** $\emptyset$
4 **while** *true* **do**
5 $V_\pi \leftarrow \{\boldsymbol{\alpha}_{\gamma_1}, \boldsymbol{\alpha}_{\gamma_2}, \dots\}$ /* Policy evaluation */
6 $\mathcal{B}_\pi \leftarrow$ the set of beliefs visited by π
7 $\pi' \leftarrow \emptyset$ /* Start policy Improvement */
8 **foreach** $b \in \mathcal{B}_\pi$ **do** /* Point-based backup */
9 $k \leftarrow$ the number of steps to reach b from b_0
10 **foreach** $a \in \mathcal{A}$ **do**
11 $\nu_a \leftarrow \emptyset$, $a_{\text{valid}} \leftarrow$ true
12 **foreach** $o \in \mathcal{O}$ **do**
13 $b_a^o \leftarrow \mathcal{T}_\mathcal{B}(b, a, o)$ /* Successor */
14 $\Gamma \leftarrow \emptyset$
15 **foreach** $\gamma \in \pi$ **do**
16 **if** $\Omega_{\gamma, b_a^o} \subseteq \Omega_\mathcal{G}$ **then** /* valid γ */
17 $\Gamma \leftarrow \Gamma \cup \{\gamma\}$
18 **if** $\Gamma = \emptyset$ **then** /* No γ is valid */
/* Invoke BPS to explore */
19 $\pi_a^o \leftarrow \text{BPS}(P, b_a^o, \mathcal{G}, h - k - 1)$
20 **if** $\pi_a^o = \emptyset$ **then** /* a is invalid */
21 $a_{\text{valid}} \leftarrow$ false, **break**
22 **else**
/* Add π_a^o for improvement */
23 $\pi' \leftarrow \pi' \cup \pi_a^o$, $\gamma_a^o \leftarrow \pi_a^o(b_a^o)$
24 **else**
25 $\gamma_a^o \leftarrow \operatorname{argmax}_{\gamma \in \Gamma} \boldsymbol{\alpha}_\gamma \cdot b_a^o$
26 $\nu_a(o) \leftarrow \gamma_a^o$ /* Record γ_a^o in ν_a */
27 **foreach** $s \in \mathcal{S}$ **do**
28 $\boldsymbol{\alpha}_a(s) \leftarrow r(s, a) + \sum_{o \in \mathcal{O}} \sum_{s' \in \mathcal{S}} \mathcal{Z}(s', a, o) \mathcal{T}(s, a, s') \boldsymbol{\alpha}_{\gamma_a^o}(s')$
29 $a \leftarrow \operatorname{argmax}_{a \in \mathcal{A} \text{ and } a_{\text{valid}}} \boldsymbol{\alpha}_a \cdot b$
30 $\gamma \leftarrow (a, \nu_a)$, $\pi' \leftarrow \pi' \cup \{\gamma\}$ /* γ is best */
31 **if** $\left(\frac{1}{|\mathcal{B}_\pi|} \sum_{b \in \mathcal{B}_\pi} (V_{\pi'}(b) - V_\pi(b)) \right) \leq \epsilon$ **then**
/* termination condition satisfied */
32 **return** π'
33 $\pi \leftarrow \pi'$

In order to satisfy the boolean objective, PBPS applies BPS to efficiently explore the goal-constrained belief space and generate a valid policy π (Line 1). If there are no valid policies within the horizon bound h, PBPS returns $\emptyset$ (Line 3). Otherwise, PBPS adapts PBPI to compute an improved policy π' with respect to the value function V_π of the current policy π. Though the goal-constrained belief space over a bounded horizon is finite, performing backup on the entire goal-constrained belief space is still costly [19]. Therefore, we perform the point-based backup [3], [5] on the set $\mathcal{B}_\pi$ of beliefs visited by the current policy π to improve efficiency.

This improved policy π' may reach a successor belief $b_a^o = \mathcal{T}_\mathcal{B}(b, a, o)$ for a belief $b \in \mathcal{B}_\pi$ (Line 13), and none of the existing conditional plans are valid starting from b_a^o (Line 18). In this case, PBPS invokes BPS to check if b_a^o is in the goal-constrained belief space (Line 19). If BPS returns $\emptyset$, there are no valid policies from the belief b_a^o, which means the action choice a is invalid for the observation o (Line 21). PBPS alternates between the computation of valid policies and policy iteration until the policy improvement $\frac{1}{|\mathcal{B}_\pi|} \sum_{b \in \mathcal{B}_\pi} (V_{\pi'}(b) - V_\pi(b))$ meets the threshold ϵ (Line 31). Note that $\mathcal{B}_\pi \subseteq \mathcal{B}_{\pi'}$ since for every $b \in \mathcal{B}_\pi$, there is one conditional plan $\gamma \in \pi'$ for b by construction and thus $b \in \mathcal{B}_{\pi'}$. Next, we describe each component (Fig. 3) of the PBPS algorithm (Algorithm 1).

Bounded Policy Synthesis: For a POMDP P, an initial belief b_0, a safe-reachability objective $\mathcal{G} = (Dest, Safe)$ and a horizon bound h, we first apply the same BPS algorithm presented in [16] to compute a valid k-step ($k \leq h$) policy π (Line 1). For completeness, we provide a brief summary of BPS. See [16] for more details. BPS first symbolically encodes the goal-constrained belief space over a bounded horizon k as a compact logical formula Φ_k, based on the encoding from Bounded Model Checking [32]. Φ_k compactly represents the requirement of reaching a goal belief safely in k steps. Then BPS computes a valid plan by checking the satisfiability of the constraint Φ_k through an SMT solver [33]. If Φ_k is satisfiable, the SMT solver returns a valid plan σ_k (the dashed green path in Fig. 2). σ_k only covers a particular observation o_i at step i. BPS tries to generate a valid policy π based on this valid plan σ_k by considering all other observations, i.e., other branches following the rectangle node at each step. If Φ_k is unsatisfiable, BPS increases the horizon and repeat the above steps until a valid policy is found or a given horizon bound is reached.

A. Policy Iteration

Once a valid k-step policy π is found, we try to improve π by adapting the PBPI algorithm presented in [4]. PBPI considers infinite-horizon policies represented as finite-state controllers while PBPS focuses on bounded-horizon policies represented as a set of conditional plans. Moreover, PBPI only deals with quantitative objectives while PBPS considers both boolean (safe-reachability) and quantitative objectives. Therefore, we cannot directly apply PBPI to compute an improved policy. Instead, we adapt the policy evaluation and the policy improvement steps in PBPI for policy iteration.

1) Policy Evaluation: For a valid k-step policy $\pi = \{\gamma_1, \gamma_2, \dots\}$, we recursively compute the $\boldsymbol{\alpha}$-vector $\boldsymbol{\alpha}_\gamma$ for each conditional plan $\gamma = (a, \nu)$ in the policy π:

- If γ is a conditional plan associated with a terminal belief, $\boldsymbol{\alpha}_\gamma(s) = r(s, a)$.

- If γ is a conditional plan associated with a non-terminal belief, we compute the $\boldsymbol{\alpha}$-vector $\boldsymbol{\alpha}_\gamma$ based on Eq. (2).

Then the value function of π can be represented as a set of $\boldsymbol{\alpha}$-vectors $V_\pi = \{\boldsymbol{\alpha}_{\gamma_1}, \boldsymbol{\alpha}_{\gamma_2}, \dots\}$ (Line 5).

2) Policy Improvement: In the policy improvement step, we compute an improved policy π' based on the value function V_π of the current policy π. This policy improvement step applies the point-based backup algorithm [3], [5] to the finite set $\mathcal{B}_\pi$ of beliefs visited by the current policy π (Line 8).

Since we consider not only a quantitative objective as in the standard point-based backup, but also a boolean objective that defines valid policies, we cannot directly apply the standard point-based backup. There are two important differences between PBPS and the standard point-based backup.

First, the standard point-based backup loops over all $\boldsymbol{\alpha}$-vectors in the value function V_π of the current policy π to find the best $\boldsymbol{\alpha}$-vector $\boldsymbol{\alpha}_a^o = \operatorname{argmax}_{\boldsymbol{\alpha} \in V_\pi} \boldsymbol{\alpha} \cdot b_a^o$ for the successor belief $b_a^o = \mathcal{T}_B(b, a, o)$. Conceptually, this step applies each conditional plan γ in the policy π to the belief b_a^o and finds the best conditional plan for b_a^o. However, not every conditional plan γ in the policy π is valid starting from the belief b_a^o. Therefore, PBPS performs an additional filtering step (Lines 15, 16, 17) to construct a subset $\Gamma \subseteq \pi$ of valid conditional plans starting from the belief b_a^o. Then PBPS selects the best conditional plan from these valid ones (Line 25).

Second, after the above filtering step, it is possible that the current policy π does not contain a valid conditional plan for the successor belief b_a^o (Line 18). In this case, PBPS invokes BPS again to compute a valid policy π_a^o for b_a^o (Line 19) and add π_a^o to the new policy π' so that in later iterations, we can improve the value of π_a^o (Line 23). By constructing this new policy π_a^o, we can explore new belief regions that have not been reached by previous policies. This exploration step together with point-based backup (when a different action becomes more optimal) expands the belief set $\mathcal{B}_\pi$. As pointed out by previous works [3]–[5], this expansion of the representative beliefs $\mathcal{B}_\pi$ is crucial to the quality of the constructed policy.

In the worst case, each policy iteration of PBPS requires $O(|\mathcal{B}_\pi||A||O|)$ calls to BPS since PBPS performs point-based backup over the representative beliefs $\mathcal{B}_\pi$ instead of the reachable set. The experiment results show (Section IV) that $\mathcal{B}_\pi$ is much smaller than the reachable belief space, which contains beliefs exponential to the horizon. Therefore, by applying point-based backup and focusing on the representative beliefs $\mathcal{B}_\pi$, PBPS significantly reduced the number of calls to BPS.

B. Algorithm Analysis

In this section, we provide two theorems to address the important properties of PBPS. Theorem 1 shows that PBPS maintains validity and keeps improving the value of the policy at each iteration before termination. Theorem 2 shows that the error introduced by PBPS due to approximation is bounded.

Theorem 1: At each iteration before termination, PBPS transforms a policy π to a new policy π' (Note that $\mathcal{B}_\pi \subseteq \mathcal{B}_{\pi'}$ as discussed at the beginning of Section III). For each belief $b \in \mathcal{B}_\pi$, the conditional plan γ specified by the new policy π' for the belief b is valid and $V_{\pi'}(b) \geq V_\pi(b)$. For at least one belief $b \in \mathcal{B}_\pi$, $V_{\pi'}(b) > V_\pi(b)$.

Proof: Each iteration of PBPS consists of two steps:

- In policy evaluation, PBPS computes the exact value function V_π of the current policy π (Line 5).
- In policy improvement, PBPS constructs a conditional plan γ for each belief $b \in \mathcal{B}_\pi$. According to Algorithm 1, for each observation o, PBPS selects the best conditional plan γ_a^o from the subset $\Gamma \subseteq \pi$ of valid conditional plans starting from the successor belief b_a^o (Line 25). When $\Gamma = \emptyset$, PBPS invokes BPS to construct a new valid conditional plan γ_a^o for the belief b_a^o (Lines 19, 23). PBPS selects the best action a for the belief b (Line 29) and construct the best conditional plan γ (Line 30) w.r.t. the value function V_π of the current policy π. By construction, γ is also valid starting from the belief b.

Thus, PBPS cannot cause a reduction in the value of any belief $b \in \mathcal{B}_\pi$ and always produces a valid policy π'. According to the termination condition (Line 31), π' improves the value for at least one belief $b \in \mathcal{B}_\pi$ before termination. ■

PBPS is an approximation method that performs point-based backup on the representative set $\mathcal{B}_\pi$ of beliefs visited by the policy π rather than the entire goal-constrained belief space $\mathcal{B}_\mathcal{G}$. As a result, PBPS implicitly prunes conditional plans for every belief $b \in \mathcal{B}_\mathcal{G} \setminus \mathcal{B}_\pi$. This implicit pruning may remove conditional plans that are part of the valid and optimal policy $\pi_\mathcal{G}^*$, producing a suboptimal policy.

Note that $\pi_\mathcal{G}^*$ is different from the optimal policy π^* without the requirement of satisfying the safe-reachability objective (π^* may be invalid). As PBPS continues improving π, the value V_π is getting closer to the value $V_{\pi_\mathcal{G}^*}$ but V_π may not converge to the optimal value V_{π^*} due to the additional safe-reachability objective. We define $\delta_\pi = \max_{b \in \mathcal{B}_\pi} (V_{\pi^*}(b) - V_\pi(b))$ to be the difference between V_{π^*} and V_π. δ_π also indicates the difference between quantitative and safe-reachability objectives. Intuitively, if we set proper rewards for goal states and unsafe states, δ_π should not be too large since the reward function also encodes the safe-reachability objectives to some extent, and the difference between quantitative and safe-reachability objectives is small. However, as discussed in [16], there exist domains where no reward function exactly encodes the safe-reachability objective and δ_π may always be greater than 0.

Theorem 2: PBPS produces a policy π with error $\eta = V_{\pi_\mathcal{G}^*}(b_0) - V_\pi(b_0) \leq h(r_{\max} - r_{\min}) d_{\mathcal{B}_\pi} + \delta_\pi$, where b_0 is the initial belief, h is the horizon bound, $r_{\max} = \max_{s,a} r(s,a)$, $r_{min} = \min_{s,a} r(s,a)$, $d_{\mathcal{B}_\pi} = \max_{b' \in \mathcal{B}_\mathcal{G}} \min_{b \in \mathcal{B}_\pi} \|b' - b\|_1$ is the maximum distance from any $b' \in \mathcal{B}_\mathcal{G}$ to $\mathcal{B}_\pi$.

Proof: Let $\gamma^* = (a^*, \nu^*)$ be the optimal conditional plan specified by $\pi_\mathcal{G}^*$ for the belief b_0 and $\gamma = (a, \nu)$ be the conditional plan specified by π for the belief b_0.

$$
\begin{aligned}
\eta &= V_{\pi_\mathcal{G}^*}(b_0) - V_\pi(b_0) = V_{\gamma^*}(b_0) - V_\gamma(b_0) \\
&= \left(\sum_{s \in \mathcal{S}} r(s, a^*) b(s) + \sum_{o \in \mathcal{O}} \Pr(o|b_0, a^*) V_{\pi_\mathcal{G}^*}(b_{a^*}^o) \right) \\
&\quad - \left(\sum_{s \in \mathcal{S}} r(s, a) b(s) + \sum_{o \in \mathcal{O}} \Pr(o|b_0, a) V_\pi(b_a^o) \right) \text{Eq. (1)} \\
&= \left(\sum_{s \in \mathcal{S}} r(s, a^*) b(s) + \sum_{o \in \mathcal{O}} \Pr(o|b_0, a^*) V_{\pi_\mathcal{G}^*}(b_{a^*}^o) \right) \\
&\quad - \left(\sum_{s \in \mathcal{S}} r(s, a) b(s) + \sum_{o \in \mathcal{O}} \Pr(o|b_0, a) V_\pi(b_a^o) \right)
\end{aligned}
$$

$$+\left(\sum_{s\in\mathcal{S}} r(s,a^*)b(s)+\sum_{o\in\mathcal{O}}\Pr(o|b_0,a^*)V_\pi(b^o_{a^*})\right)$$

$$-\left(\sum_{s\in\mathcal{S}} r(s,a^*)b(s)+\sum_{o\in\mathcal{O}}\Pr(o|b_0,a^*)V_\pi(b^o_{a^*})\right) \text{ add } 0$$

$$=\sum_{o\in\mathcal{O}}\Pr(o|b_0,a^*)\left(V_{\pi^*_{\mathcal{G}}}(b^o_{a^*})-V_\pi(b^o_{a^*})\right)$$

$$+\left(\sum_{s\in\mathcal{S}} r(s,a^*)b(s)+\sum_{o\in\mathcal{O}}\Pr(o|b_0,a^*)V_\pi(b^o_{a^*})\right)$$

$$-\left(\sum_{s\in\mathcal{S}} r(s,a)b(s)+\sum_{o\in\mathcal{O}}\Pr(o|b_0,a)V_\pi(b^o_{a})\right) \quad (3)$$

Since π specifies γ for the belief b_0, γ should be best for b_0 w.r.t. the value function V_π. Thus the last two terms in Eq. (3): $\left(\sum_{s\in\mathcal{S}} r(s,a^*)b(s)+\sum_{o\in\mathcal{O}}\Pr(o|b_0,a^*)V_\pi(b^o_{a^*})\right)-\left(\sum_{s\in\mathcal{S}} r(s,a)b(s)+\sum_{o\in\mathcal{O}}\Pr(o|b_0,a)V_\pi(b^o_{a})\right)\le 0$, and $\eta\le\sum_{o\in\mathcal{O}}\Pr(o|b_0,a^*)\left(V_{\pi^*_{\mathcal{G}}}(b^o_{a^*})-V_\pi(b^o_{a^*})\right)$. Let $b'\in\bigcup_{o\in\mathcal{O}}\{b^o_{a^*}\}$ be the successor where PBPS makes its worst error, $\gamma^*_o\in\pi^*_{\mathcal{G}}$ be the optimal conditional plan for b', $\gamma_o\in\pi$ be the best conditional plan for b' and $b\in\mathcal{B}_\pi$ be the belief associated with γ_o. Then

$$\eta\le\sum_{o\in\mathcal{O}}\Pr(o|b_0,a^*)(\boldsymbol{\alpha}_{\gamma^*_o}\cdot b'-\boldsymbol{\alpha}_{\gamma_o}\cdot b')$$

$$\le\boldsymbol{\alpha}_{\gamma^*_o}\cdot b'-\boldsymbol{\alpha}_{\gamma_o}\cdot b'\sum_{o\in\mathcal{O}}\Pr(o|b_0,a^*)=1$$

$$=(\boldsymbol{\alpha}_{\gamma^*_o}-\boldsymbol{\alpha}_{\gamma_o})\cdot(b'-b)+(\boldsymbol{\alpha}_{\gamma^*_o}-\boldsymbol{\alpha}_{\gamma_o})\cdot b \text{ add } 0 \quad (4)$$

Following the derivations in [3], for the first term in Eq. (4)

$$(\boldsymbol{\alpha}_{\gamma^*_o}-\boldsymbol{\alpha}_{\gamma_o})\cdot(b'-b)$$
$$\le\|\boldsymbol{\alpha}_{\gamma^*_o}-\boldsymbol{\alpha}_{\gamma_o}\|_\infty\|b'-b\|_1 \text{ Holder}'s\text{inequality}$$
$$\le\|\boldsymbol{\alpha}_{\gamma^*_o}-\boldsymbol{\alpha}_{\gamma_o}\|_\infty d_{\mathcal{B}_\pi} \text{ definition of } d_{\mathcal{B}_\pi}$$
$$\le h(r_{\max}-r_{\min})d_{\mathcal{B}_\pi}$$

The last inequality holds since $\boldsymbol{\alpha}$-vectors represent the reward starting from some state within the horizon bound h. For the second term in Eq. (4), since γ^*_o may be invalid starting from b, $\boldsymbol{\alpha}_{\gamma^*_o}\cdot b\le V_{\pi^*}(b)$ where π^* is an optimal policy without the requirement of satisfying the safe-reachability objective and may be invalid. According to the definition of δ_π, $(\boldsymbol{\alpha}_{\gamma^*_o}-\boldsymbol{\alpha}_{\gamma_o})\cdot b\le\delta_\pi$. Therefore, $\eta\le h(r_{\max}-r_{\min})d_{\mathcal{B}_\pi}+\delta_\pi$. ■

As we expand the set $\mathcal{B}_\pi$ through the exploration step (Line 19) and point-based backup, $\mathcal{B}_\pi$ covers more beliefs from the goal-constrained belief space $\mathcal{B}_{\mathcal{G}}$. Therefore, the first term $h(r_{\max}-r_{\min})d_{\mathcal{B}_\pi}$ in the error bound converges to 0 since the distance $d_{\mathcal{B}_\pi}$ converges to 0. As discussed before, the value of the second term δ_π in the error bound is closely related to the reward function. How to design a proper reward function that minimizes δ_π is beyond the scope of this work.

IV. Experiments

We test PBPS on the kitchen domain [16] (horizon bound $h=20$) and the Tag domain [3] ($h=30$). We also validate PBPS on a Fetch robot for the domain shown in Fig. 1 ($h=20$). We use Z3 [33] as our SMT solver. All experiments were conducted on a 2.9 GHz Intel processor with 16 GB memory.

TABLE I
Performance of PBPS With and Without the Exploration Step for Different Problems

Domain	Reward		Time (s)		$\lvert\mathcal{B}_\pi\rvert$	
	w. exp.	no exp.	w. exp.	no exp.	w. exp.	no exp.
Kitchen ($M=1$)	-9.350	-9.350	18.166	10.321	147	94
Kitchen ($M=2$)	-13.040	-24.296	347.281	33.798	396	73
Kitchen ($M=3$)	-16.882	-31.801	3611.684	290.019	474	64
Tag ($\delta=0.5$)	-6.987	-9.070	509.235	21.352	370	109
Tag ($\delta=0.6$)	-6.871	-9.108	1132.996	21.985	811	107

Kitchen Domain: In the kitchen domain [16], a robot needs to pick up a cup from the storage while avoiding collisions with M uncertain obstacles. This domain is an example scenario where we desire a correctness guarantee of accomplishing tasks, and POMDPs with boolean objectives offer a better guarantee than quantitative POMDP models [16].

The kitchen is discretized into $N=36$ regions. The actuation and perception of the robot are imperfect, modeled as ten uncertain actions: *move* and *look* in four directions, pick-up using the left hand and pick-up using the right hand. The robot starts at a known initial location. However, due to the robot's imperfect perception, the location of the robot and the locations of uncertain obstacles are all partially observable during execution. The number of states in the kitchen domain is $|\mathcal{S}|=C(N,M)\cdot N$, where $C(N,M)$ is the number of M-combinations from the set of N regions. In the largest test ($M=3$) there are more than 10^5 states. We use the same safe-reachability objective $\mathcal{G}=(Dest,Safe)$ as in [16]:

$$Dest=\{b\in\mathcal{B}|\sum b(\text{target cup in hand})>1-\delta_1\}$$

$$Safe=\{b\in\mathcal{B}|\sum b(\text{robot in collision})<\delta_2\}$$

where δ_1 and δ_2 are small values that represent tolerance.

For the quantitative objective, the robot should avoid certain regions if possible. We assign a reward of -10 for states where the robot is in these regions and a reward of -1 for each action.

Tag Domain: In the Tag domain [3], the task for the robot is to search for and tag a moving agent in a grid with 29 locations. The agent follows a fixed strategy that intentionally moves away from the robot. Both the robot and the agent can move in four directions or stay. The robot's location is fully observable while the agent's location is unobservable unless the robot and the agent are in the same location. The safe-reachability objective $\mathcal{G}=(Dest,Safe)$ we specify for this domain is: (1) $Dest$ contains beliefs where the robot can tag the agent with a probability at least δ; (2) $Safe=\mathcal{B}$ since there are no unsafe states and all beliefs are safe beliefs.

Results: The performance results of PBPS are summarized in Table I, which shows the rewards achieved by following the policies, the computation time and the size (the number of beliefs) of the representative belief set $|\mathcal{B}_\pi|$. To evaluate how the quality of constructed policies is affected by the exploration step (Line 19) of PBPS (Algorithm 1) where we invoke BPS to explore new belief regions, we test PBPS in two settings: with and without the exploration step.

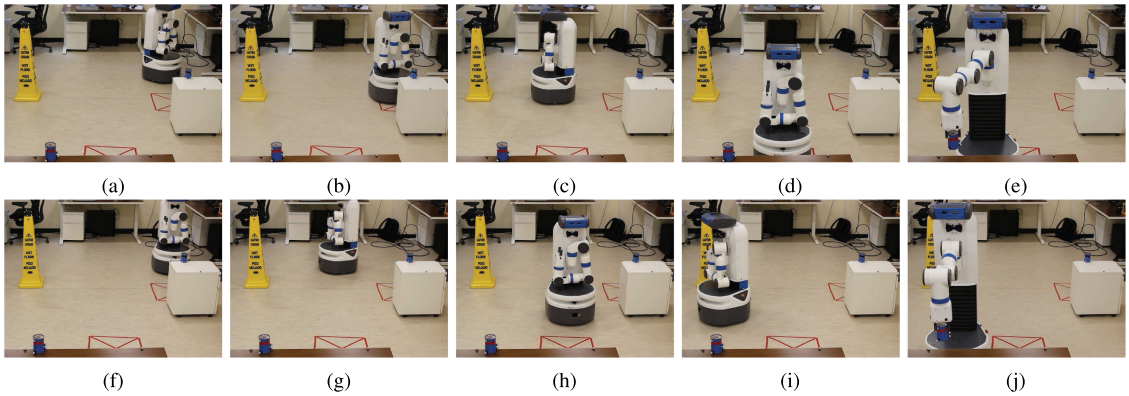

Fig. 4. Executions of policies constructed by BPS (first row) and PBPS (second row) for the domain shown in Fig. 1.

As we can see from Table I, PBPS with exploration achieves much better reward compared to PBPS without exploration for large problems. However, this exploration step is expensive and requires more computation time. The results also demonstrate the advantage of approximate policies against exact policies: computing exact policies requires exhaustively exploring the whole goal-constrained belief space, which is intractable in practice since exploration is costly as shown in Table I. On the contrary, PBPS produces approximate policies by performing the point-based backup, and only explores belief regions of the goal-constrained belief space that are promising based on the current value function. Moreover, as we can see from the experiment results, $\mathcal{B}_\pi$ for the tests only contains hundreds of beliefs, while the reachable belief space contains beliefs exponential to the planning horizon. Therefore, by applying point-based backup and focusing on the representative belief set $\mathcal{B}_\pi$, PBPS significantly reduced the number of calls to BPS.

Comparison with Existing POMDP methods: Previous work [15] on POMDPs with boolean and quantitative objectives are restricted to *almost-sure satisfaction*. The domains in the experiments cannot be modeled as almost-sure satisfaction problems due to the probability threshold constraints in the boolean objectives. The objectives of the tested domains can be implicitly encoded as POMDPs with only quantitative objectives (unconstrained POMDPs) or C/RS/CC-POMDPs. Our previous work [16] have shown that, in certain domains such as the kitchen domain in our experiments, using the implicit representations (unconstrained/C/RS/CC-POMDPs) does not provide the same guarantee as using our formulation of POMDPs with explicit boolean and quantitative objectives.

To evaluate the policy quality generated by PBPS, we run a state-of-the-art POMDP solver SARSOP [5] on the Tag domain. The policy produced by SARSOP achieves an expected reward of -6.02 but only visits beliefs where the probability of tagging the agent is at most 0.539. The policy generated by PBPS for the Tag domain with $\delta = 0.6$ guarantees that the robot eventually visits a belief where the probability of tagging the agent is at least 0.6. For the kitchen domain, since PBPS considers both safe-reachability and quantitative objectives, PBPS offers a better guarantee of accomplishing tasks than solvers for quantitative POMDP models as discussed in [16].

Physical Validation: We validate PBPS on a Fetch robot for the domain in Fig. 1. The setup of this domain is similar to the kitchen domain. The Fetch needs to pick up the blue can on the table while avoiding collisions with uncertain obstacles such as floor signs and file cabinets, which can be placed in different locations. There are two regions marked with red tape that the robot should avoid if possible. We assign the reward of -10 for states where the robot is in these regions. We also assign a reward of -1 for each action.

The POMDP's state space consists of the locations of robot and object. We use a Vicon system to detect object locations, which is usually accurate but can still produce false negative and false positive due to occlusion or inappropriate Vicon marker configurations on objects. We estimate the false negative and false positive probabilities by counting the false negative and false positive events during 100 Vicon detections. The POMDP's probabilistic observation function is defined based on the false negative and false positive probabilities. Sometimes the Fetch may fail to move its base when given a *move* action command and stay in the same place. We estimate the failure probability of these move actions by counting the failure events during 100 *move* action executions. The POMDP's probabilistic transition function is defined based on this failure probability.

We evaluate BPS and PBPS in this office domain. Fig. 4 a, 4 b, 4 c, 4 d, and 4 e show the execution of the policy constructed by BPS. Fig. 4 f, 4 g, 4 h, 4 i, and 4 j show the execution of the policy constructed by PBPS. As shown in these figures, both executions accomplish the task safely. However, the execution for BPS visits both red regions (Fig. 4 b and 4 d) that the robot should avoid while the execution for PBPS visits zero red regions. Therefore, PBPS produces a policy that is more optimal than that produced by BPS by considering both boolean and quantitative objectives. We also run the physical experiment on the Fetch ten times. We notice that in one run execution failure happens. The cause of the failure is a false negative observation of the wet-floor sign resulting in an unsafe state, which is rare but could still happen due to uncertain perception. However, the probability of visiting unsafe states is still below the given threshold, thanks to the guarantee of PBPS.

V. Discussion

We presented a new policy synthesis approach called PBPS for POMDPs with both safe-reachability and quantitative objectives. Our approach combines BPS [16] and Point-Based Policy Iteration [4]: we apply BPS to efficiently explore the goal-constrained belief space and perform the point-based backup on a finite subset of representative beliefs from the goal-constrained belief space to compute good approximate policies

that satisfy safe-reachability objectives. We prove that PBPS maintains validity and guarantees improved policies at each iteration before termination. Moreover, we prove that the error introduced by PBPS is bounded. Both simulation and physical experiments demonstrate that the policies constructed by PBPS achieve the safe-reachability objective and are of high quality with respect to the quantitative objective.

Our approach assumes a discretized environment and a bounded horizon, and focuses on safe-reachability objectives. These assumptions are likely to hold in certain domains where we want the robot to safely accomplish the task in finite time, e.g., the corridor domain in [34] and the underwater navigation problem in [5]. While many robot tasks can be specified using safe-reachability objectives, there are settings in robotics such as patrolling and surveillance that require general temporal properties to specify the tasks. Investigating how to extend PBPS for general temporal properties is a promising future direction. Another important ongoing question is how to extend PBPS for continuous POMDPs [8]–[11].

REFERENCES

[1] R. D. Smallwood and E. J. Sondik, "The optimal control of partially observable Markov processes over a finite horizon," *Operations Res.*, vol. 21, no. 5, pp. 1071–1088, 1973.

[2] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, no. 1/2, pp. 99–134, May 1998.

[3] J. Pineau, G. Gordon, and S. Thrun, "Point-based value iteration: An anytime algorithm for POMDPs," in *Proc. Int. Joint Conf. Artif. Intell.*, 2003, pp. 1025–1030.

[4] S. Ji, R. Parr, H. Li, X. Liao, and L. Carin, "Point-based policy iteration," in *Proc. 22nd Nat. Conf. Artif. Intell.*, 2007, pp. 1243–1249.

[5] H. Kurniawati, D. Hsu, and W. S. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," in *Proc. Robot. Sci. Syst.*, 2008.

[6] Y. Luo, H. Bai, D. Hsu, and W. S. Lee, "Importance sampling for online planning under uncertainty," in *Proc. WAFR*, 2016.

[7] P. Cai, Y. Luo, D. Hsu, and W. S. Lee, "HyP-DESPOT: A hybrid parallel algorithm for online planning under uncertainty," in *Proc. Robot. Sci. Syst.*, 2018.

[8] J. M. Porta, N. Vlassis, M. T. J. Spaan, and P. Poupart, "Point-based value iteration for continuous POMDPs," *Jo. Mach. Learn. Res.*, vol. 7, pp. 2329–2367, Dec. 2006.

[9] J. Hoey and P. Poupart, "Solving POMDPs with continuous or large discrete observation spaces," in *Proc. Int. Joint Conf. Artif. Intell.*, 2005, pp. 1332–1338.

[10] H. Bai, D. Hsu, W. S. Lee, and V. A. Ngo, "Monte Carlo value iteration for continuous-state POMDPs," in *Proc. WAFR*, 2010, pp. 175–191.

[11] K. M. Seiler, H. Kurniawati, and S. P. N. Singh, "An online and approximate solver for POMDPs with continuous action space," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 2290–2297.

[12] M. Svoreňová *et al.*, "Temporal logic motion planning using POMDPs with parity objectives: Case study paper," in *Proc. HSCC*, 2015, pp. 233–238.

[13] K. Chatterjee, M. Chmelík, R. Gupta, and A. Kanodia, "Qualitative analysis of POMDPs with temporal logic specifications for robotics applications," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 325–330.

[14] K. Chatterjee, M. Chmelík, and J. Davies, "A symbolic SAT-based algorithm for almost-sure reachability with small strategies in POMDPs," in *Proc. AAAI*, 2016, pp. 3225–3232.

[15] K. Chatterjee, M. Chmelík, R. Gupta, and A. Kanodia, "Optimal cost almost-sure reachability in POMDPs," *Artif. Intell.*, vol. 234, no. C, pp. 26–48, May 2016.

[16] Y. Wang, S. Chaudhuri, and L. E. Kavraki, "Bounded policy synthesis for POMDPs with safe-reachability objectives," in *AAMAS*, 2018, pp. 238–246.

[17] Y. Wang, S. Chaudhuri, and L. E. Kavraki, "Online partial conditional plan synthesis for POMDPs with safe-reachability objectives," in *Proc. WAFR*, 2018.

[18] A. Undurti and J. P. How, "An online algorithm for constrained POMDPs," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2010, pp. 3966–3973.

[19] L. Valiant, "The complexity of enumeration and reliability problems," *SIAM J. Comput.*, vol. 8, no. 3, pp. 410–421, 1979.

[20] J. D. Isom, S. P. Meyn, and R. D. Braatz, "Piecewise linear dynamic programming for constrained POMDPs," in *Proc. AAAI*, 2008, pp. 291–296.

[21] D. Kim, J. Lee, K. Kim, and P. Poupart, "Point-based value iteration for constrained POMDPs," in *Proc. Int. Joint Conf. Artif. Intell.*, 2011, pp. 1968–1974.

[22] P. Poupart, A. Malhotra, P. Pei, K. Kim, B. Goh, and M. Bowling, "Approximate linear programming for constrained partially observable Markov decision processes," in *Proc. AAAI*, 2015, pp. 3342–3348.

[23] P. Santana, S. Thiébaux, and B. Williams, "RAO*: An algorithm for chance-constrained POMDP's," in *Proc. AAAI*, 2016, pp. 3308–3314.

[24] J. Marecki and P. Varakantham, "Risk-sensitive planning in partially observable environments," in *Proc. AAMAS*, 2010, pp. 1357–1368.

[25] P. Hou, W. Yeoh, and P. Varakantham, "Solving risk-sensitive POMDPs with and without cost observations," in *Proc. AAAI*, 2016, pp. 3138–3144.

[26] E. A. Hansen, "Solving POMDPs by searching in policy space," in *Proc. UAI*, 1998, pp. 211–219.

[27] C. Papadimitriou and J. N. Tsitsiklis, "The complexity of Markov decision processes," *Math. Operations Res.*, vol. 12, no. 3, pp. 441–450, Aug. 1987.

[28] A. Paz, *Introduction to Probabilistic Automata*. Orlando, FL, USA: Academic Press, Inc., 1971.

[29] O. Madani, S. Hanks, and A. Condon, "On the undecidability of probabilistic planning and related stochastic optimization problems," *Artif. Intell.*, vol. 147, no. 1/2, pp. 5–34, Jul. 2003.

[30] K. Chatterjee, M. Chmelík, and M. Tracol, "What is decidable about partially observable Markov decision processes with omega-regular objectives," *J. Comput. Syst. Sci.*, vol. 82, no. 5, pp. 878–911, Aug. 2016.

[31] M. Mundhenk, J. Goldsmith, C. Lusena, and E. Allender, "Complexity of finite-horizon Markov decision process problems," *J. ACM*, vol. 47, no. 4, pp. 681–720, Jul. 2000.

[32] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, and Y. Zhu, "Bounded model checking," *Adv. Comput.*, vol. 58, pp. 117–148, 2003.

[33] L. De Moura and N. Bjørner, "Z3: An efficient SMT solver," in *TACAS*, C. R. Ramakrishnan and J. Rehof, Eds. Berlin: Springer-Verlag, 2008, pp. 337–340.

[34] D. Hadfield-Menell, E. Groshev, R. Chitnis, and P. Abbeel, "Modular task and motion planning in belief space," in *Proc. IROS*, Sep. 2015, pp. 4991–4998.