Speeding up POMDP Planning via Simplification

Ori Sztyglic[1] and Vadim Indelman[2]
[1]Department of Computer Science [2]Department of Aerospace Engineering
Technion - Israel Institute of Technology, Haifa 32000, Israel
ori.sztyglic@gmail.com, vadim.indelman@technion.ac.il

***Abstract*— In this paper, we consider online planning in partially observable domains. Solving the corresponding POMDP problem is a very challenging task, particularly in an online setting. Our key contribution is a novel algorithmic approach, *Simplified Information Theoretic Belief Space Planning* (SITH-BSP), which aims to speed up POMDP planning considering belief-dependent rewards, without compromising the solution's accuracy. We do so by mathematically relating the simplified elements of the problem to the corresponding counterparts of the original problem. Specifically, we focus on belief simplification and use it to formulate bounds on the corresponding original belief-dependent rewards. These bounds in turn are used to perform branch pruning over the belief tree, in the process of extracting the optimal policy from this existing belief tree. We further introduce the notion of adaptive simplification, while re-using calculations between different simplification levels, and exploit it to prune, at each level in the belief tree, all branches but one. Therefore, our approach is guaranteed to find the optimal solution (policy) that corresponds to the given belief tree but with substantial speedup. As a second key contribution, we derive novel analytical bounds for differential entropy, considering a sampling-based belief representation, which we believe are of interest on their own. We validate our approach in simulation using these bounds and where simplification corresponds to reducing the number of samples, exhibiting a significant computational speedup while yielding the optimal solution for the given belief tree.**

I. Introduction

In the world of autonomous agents operating in an uncertain environment, a Partially Observable Markov Decision Process (POMDP) provides a principled mathematical framework for planning under uncertainty. Solving a POMDP is proven to be PSPACE-Complete [1] giving rise to many sub-optimal approaches that trade-off optimality and runtime complexity. This difficulty is more keenly felt when considering an online-setting of such autonomous tasks, i.e. when a robot has few seconds in every time step to execute the action it deems to be 'optimal'.

In a partially observable setting, the robot maintains a *belief*, a posterior distribution over the state of interest, given actions and observations history the robot has executed and gathered so far. At each planning session, given this belief the robot determines the optimal policy (or action sequence) by constructing and traversing a belief tree, as illustrated in Fig. 1, which models how the POMDP can evolve into the future considering some finite horizon of time steps. When constructing the tree in planning time, the tree branches when taking an action and again when acquiring an observation.

This research was supported by the Israel Science Foundation (ISF) and by a donation from the Zuckerman Fund to the Technion Center for Machine Learning and Intelligent Systems (MLIS).

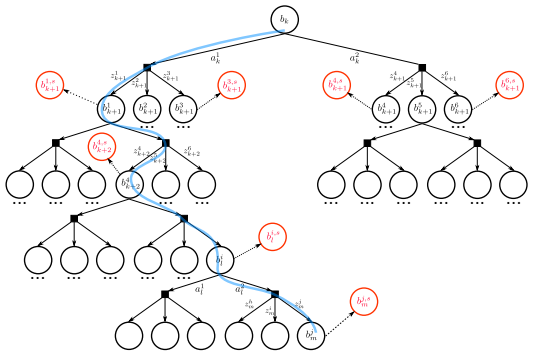


Fig. 1: For each belief node b in a given belief tree a simplified belief b^s is calculated and used to formulate bounds over the corresponding belief-dependent reward. These bounds are then used to prune branches while calculating the optimal policy.

This setting presents two main difficulties known as the *curse of dimensionality* and the *curse of history*. These two problems gave rise to many works trying to reduce computation time when solving the POMDP such as [2] and [3]. In recent years many works began to address the POMDP setting with more 'real-world' settings, such as continuous or huge observation and action spaces e.g. [4]–[6]. Some recent works consider information-theoretic rewards (e.g. [7], [8]).

Information theoretic rewards, such as differential entropy and information gain, are essential in various robotics tasks such informative planning, active estimation, active SLAM, and efficient sensing (see e.g. [9], [10]). However, these rewards can also be very computationally expensive, compared to rewards over the state, since they may require to reconstruct the manifold representing the belief and perform expensive integration operations. Getting back to planning, this enhanced computational burden is incurred for *all* nodes in the belief tree.

In this work, we consider this challenging setting of continuous state and observation spaces and information-theoretic rewards. Before stating our actual contributions we review the most relevant works in this context.

II. Related Work

Accurately solving huge (or continuous) state and observations spaces POMDP is time-consuming. Early methods, tackling MDPs with huge state space, such as [11] build the belief tree up to a predetermined planning horizon. Next, they choose the optimal action at the root by utilizing the

Bellman operator from the leaves up to the root of the tree, updating needed estimations along the way. The purpose is to avoid iterating the entire belief space and only consider belief elements that are achievable via sampled actions-observations sequences executed from the root of the tree. However, building the tree in full is still highly expensive. [2] introduced POMCP, an algorithm that applies Monte Carlo Tree Search over the POMDP's equivalent Belief-MDP. This method and its numerous expansions ([4], [6], [12] etc.) avoid building the full belief tree. They do so by building it incrementally, revealing only the "promising" parts. Commonly, they make use of a strategy to balance exploration and exploitation, such as UCB [13]. However, most of these algorithms are not suitable for information-theoretic rewards since they require the belief to be represented as a complete set of particles. A demand that can be hardly met throughout the tree when the observation space is continuous because they each time simulate only a single state sample (particle). Exceptions are PFT-DPW [4] and IPFT [7] algorithms. These algorithms represent the belief nodes as a set of particles and each time a belief node is added to the tree, they propagate all the particles using a particle filter.

Another paradigm meant to speed up planning is the use of upper and lower bounds throughout the tree nodes [14]. The gap induced by the bounds is used as a heuristics to choose "promising" sections of the tree to expand and when the bounds at the root are close enough they serve as stopping criteria to the algorithm. Smith and Simmons presented HSVI [3], which is an early seminal work that makes use of bounds over the belief tree in the context of POMDP planning. However, their approach is not suitable for the setting we consider in this work. State-of-the-art algorithm DESPOT [15] and its extension DESPOT-α [5] are also inadequate for information-theoretic rewards. The former, like others, simulates a single particle at a time which can lead to belief nodes containing a single particle; the latter makes use of α-vectors and therefore assumes a specific rewards structure that is not general enough and does not settle with information-theoretic rewards.

It is worth mentioning additional methods such as belief compression can speed up planning [16]. Yet, these kinds of methods are driven by error minimization such as belief representation error induced by the relaxation they carry out. In turn, these errors may result in a sub-optimal solution. On the other hand, simplification [17], [18] strives to perform relaxation of the decision-making problem while assuring the same solution as the original (non-relaxed) problem. When it is not feasible, the potential objective error (of executing one action over the optimal one) is bounded as part of the simplification scheme.

III. Contribution

All of the mentioned approaches practice one or more of the celebrated methods meant to speed up planning. However, nearly all of them are not suitable for information-theoretic rewards where the source of the computational burden shifts towards the reward calculation. Our method addresses this challenge directly by utilizing a notion called *Simplification*. We derive a new algorithmic approach, *Complementary* to existing POMDP planning algorithms. The approach is meant to speed up planning, considering information-theoretic rewards, non-parametric (sample-based) belief representation, and when planning is done by traversing the belief tree. Our method follows the general sparse sampling planning scheme and thus lays the foundations to expanding it to additional planning techniques. Further, to demonstrate our approach we derive novel bounds over the particle-based approximation of the differential entropy. The bounds are easy to calculate, converge to the actual entropy approximation on-demand, and can be efficiently updated incrementally. Subsequently, our approach demonstrates substantial speedup while securing an identical to the baseline solution when tested on a continuous state and observation setting and using the differential entropy approximation as a reward function. This paper is accompanied with supplementary material [19].

IV. Background

We model the Partially Observable Markov Decision Process (POMDP) for the finite horizon case, as a 7-tuple: $M = (\mathcal{X}, \mathcal{A}, T, r, \mathcal{Z}, \mathcal{O}, b_0)$, where $\mathcal{X}, \mathcal{A}$ and $\mathcal{Z}$ are the state, action and observation spaces, respectively. $T(x, a, x') \triangleq \mathbb{P}(x' \mid x, a)$ is the probabilistic transition model from past state $x \in \mathcal{X}$ to state $x' \in \mathcal{X}$ via action $a \in \mathcal{A}$. $\mathcal{O}(x, z) \triangleq \mathbb{P}(z \mid x)$ is the observation model expressing the measurement likelihood $z \in \mathcal{Z}$ for a given state. b_0 is the initial belief we have on the state at planning time. The *belief* is a posterior distribution over the state given all actions and measurements so far. It can be updated recursively via Bayes rule as $b[x'] = \eta \int \mathbb{P}(z' \mid x')\mathbb{P}(x' \mid x, a)b[x]dx$, where η is a normalization constant.

In this paper we consider a belief-dependent reward function $r(b, a)$. It allows one to use information-theoretic costs such as (differential) entropy, information gain and mutual information, thereby reasoning about future posterior uncertainty within the decision making process.

We denote the posterior belief at planning time k as $b[x_k] \triangleq \mathbb{P}(x_k \mid a_{0:k-1}, z_{1:k})$. Further, we denote by π_{k+j} a policy for time step $k + j$, i.e. $\pi_{k+j}(b[x_{k+j}])$ determines the action a_{k+j}. Let $\pi_{k+} \triangleq \pi_{k:k+L-1}$ represent a sequence of policies for the entire planning horizon of L steps that starts at time instant k. To shorten notations, we shall also use in the sequel $\pi_{(k+j)+} \triangleq \pi_{k+j:k+L-1}$, as well as $b_{k+j} \triangleq b[x_{k+j}]$. When solving a POMDP, one is trying to find the optimal policy that maximizes the objective (value) function,

$$J(b_k, \pi_{k+}) = \mathop{\mathbb{E}}_{z_{k+1:k+L}} \{ \sum_{i=k}^{k+L-1} r(b_i, \pi_i(b_i)) + r(b_{k+L}) \}, \quad (1)$$

where $r(b_{k+L})$ is the terminal reward. We may also consider a more general reward structure of $r(b_i, b_{i-1}, a_i)$, which is required, for example, to support information-theoretic reward functions such as information gain, and a specific

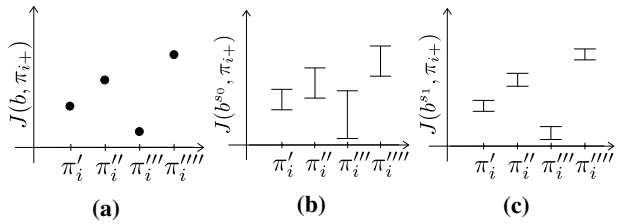


Fig. 2: Action elimination using bounds. **(a)** Objective for some belief b and candidate policies; **(b)** Objective bounds given belief is simplified to level s_0; **(c)** Objective bounds given belief is simplified to level s_1.

sampling based approximation of differential entropy [20] that we shall use in Section V-C.2. As earlier, action a_i is determined by $\pi_i(b_i)$. The optimal policy $\pi^\star_{k+} \triangleq \pi^\star_{k:k+L-1}$ and the corresponding objective function are given by

$$\pi^\star_{k+} = \arg\max_{\pi_{k+}} J(b_k, \pi_{k+}),\ J^\star(b_k) = \max_{\pi_{k+}} J(b_k, \pi_{k+}). \quad (2)$$

Further, the objective function (1) can be written recursively, i.e., the Bellman equation.

$$J(b_k, \pi_{k+}) = r(b_k, a_k) + \mathop{\mathbb{E}}_{z_{k+1}} \{J(b_{k+1}, \pi_{(k+1)+})\}. \quad (3)$$

In this paper, we resort to branch-and-bound pruning to speed up planning over the belief tree. Instead of calculating rewards (Fig. 2a), we calculate the bounds over them (Fig. 2b), and if they are not tight enough we tighten them up (Fig. 2c) so we can prune tree branches.

V. Our Approach

A. Simplification

Simplification is any sort of relaxation of the POMDP elements (components of M). In this paper we consider a specific instantiation of this general simplification framework; namely, we suggest to use a simplified belief b^s to bound the reward instead of calculating it in full. Note, in this setting, simplification does not impact the distribution over which the expectation is taken. Thus, a given belief tree of the original problem corresponds also to the simplified problem. We assume the belief tree was built in some manner and it is given. This setting settles well with *Sparse Sampling* approaches [11] which are extremely general and widely used. Hence, we will derive our approach over a similar setting. Expansions to additional approaches such as MCTS are possible and constitute an active research field [21].

As mentioned, we aim to simplify the reward calculations. Namely, the original reward model is bounded using the simplified belief and takes the form

$$\mathbf{lb}(b^s, b, a) \le r(b, a) \le \mathbf{ub}(b^s, b, a), \quad (4)$$

where $\mathbf{lb}$ and $\mathbf{ub}$ are the corresponding lower and upper bounds, respectively. A key requirement is reduced computational complexity of these bounds compared to the complexity of the original reward. Furthermore, our formulation can be extended straightforwardly to support also information-theoretic rewards of the form $r(b_{i-1}, b_i)$, which involve two (consecutive) beliefs b_{i-1} and b_i, such as information gain. In such a case, the corresponding bounds would be

$$\mathbf{lb}(b^s_{i-1}, b^s_i, b_{i-1}, b_i) \le r(b_{i-1}, b_i) \le \mathbf{ub}(b^s_{i-1}, b^s_i, b_{i-1}, b_i). \quad (5)$$

In this section we formulate our approach considering the general form of the bounds (4). In Section V-C.2 we derive novel bounds of the form (5).

Given a belief tree of the original problem M, Instead of calculating the expensive reward $r(b, a)$ for each belief node b in this belief tree, we first calculate the corresponding simplified belief b^s, as illustrated in Fig. 1, and then formulate the bounds $\mathbf{lb}$ and $\mathbf{ub}$ from (5). Moreover, we can now traverse the belief tree bottom-up and calculate recursively bounds over the objective (value) function at each node b_i via Bellman equation (3) as described below for $i \in [k, k+L-1]$.

$$\begin{aligned} \mathcal{UB}(b_i, \pi_{i+}) &= \mathbf{ub}(b^s_i, b_i, a)) + \mathop{\mathbb{E}}_{z_{i+1}} \{\mathcal{UB}(b_{i+1}, \pi_{(i+1)+})\} \\ \mathcal{LB}(b_i, \pi_{i+}) &= \mathbf{lb}(b^s_i, b_i, a) + \mathop{\mathbb{E}}_{z_{i+1}} \{\mathcal{LB}(b_{i+1}, \pi_{(i+1)+})\}, \end{aligned} \quad (6)$$

with $\pi_{i+} = \{\pi_i, \pi_{(i+1)+}\}$ and $a = \pi_i(b_i) \in \mathcal{A}$, and where the expectation is taken with respect to $\mathbb{P}(\cdot \mid b_i, a)$, and the bounds are initialized at the terminal rewards (Lth time step in the planning horizon) as $\mathcal{LB}(b_{k+L}) = \mathbf{lb}(r^s(b_{k+L}))$ and $\mathcal{UB}(b_{k+L}) = \mathbf{ub}(r^s(b_{k+L}))$. This recursive procedure is common and practiced in many works (e.g. [15]), yet a key difference is that our bounds (5) are obtained by relating the simplified POMDP elements to the original problem. Eq. (6) is a recursive update considering *some* trajectory down the belief tree determined by some policy π_{i+}. In contrast, we now consider upper and lower bounds for the optimal policy $\pi^\star_{i+}$. We denote these bounds as

$$\mathcal{UB}^\star(b_i) \triangleq \mathcal{UB}(b_i, \pi^\star_{i+}), \quad \mathcal{LB}^\star(b_i) \triangleq \mathcal{LB}(b_i, \pi^\star_{i+}). \quad (7)$$

Updating the bounds (7) is done recursively in two steps. First by considering the expansion of the already-calculated bounds $\mathcal{UB}^\star(b_{i+1})$ and $\mathcal{LB}^\star(b_{i+1})$ via (6).

In practice, the expectation over observations is approximated by a parametric number of samples, n_z, which yields

$$\begin{aligned} \mathcal{UB}(b_i, \{a, \pi^\star_{(i+1)+}\}) &= \mathbf{ub}(b^s_i, b_i, a) + \frac{1}{n_z} \sum_l \mathcal{UB}^\star(b^l_{i+1}) \\ \mathcal{LB}(b_i, \{a, \pi^\star_{(i+1)+}\}) &= \mathbf{lb}(b^s_i, b_i, a) + \frac{1}{n_z} \sum_l \mathcal{LB}^\star(b^l_{i+1}) \end{aligned} \quad (8)$$

where superscript l is the belief node index corresponding to the z^l observation. The above is defined for each $a \in \mathcal{A}$.

Second, we perform branch pruning using Alg. 2 and as explained in Section IV: For each action $a \in \mathcal{A}$ we have corresponding bounds acquired via (8). For sufficiently tight bounds, all branches but one can be pruned (w.l.o.g. the branch corresponding to action $a^\star \in \mathcal{A}$). Thus, the bounds corresponding to action $a^\star$ hold: $\mathcal{UB}^\star(b_i) = \mathcal{UB}(b_i, \{a^\star, \pi^\star_{(i+1)+}\}), \mathcal{LB}^\star(b_i) = \mathcal{LB}(b_i, \{a^\star, \pi^\star_{(i+1)+}\})$, and $\pi^\star_{i+}(b_i) = \{a^\star, \pi^\star_{(i+1)+}\}$, resulting upper and lower bounds on the *optimal* objective (value) function $J^\star(b_i)$,

$$\mathcal{LB}^\star(b_i) \le J^\star(b_i) \le \mathcal{UB}^\star(b_i), \quad (9)$$

and the optimal policy $\pi^\star_{i+}(b_i)$. See illustration in Fig. 3b.

Yet, this formulation presents a difficulty. It is generally not guaranteed that after using Alg. 2 we are left with a single branch in each belief node since the bounds might overlap

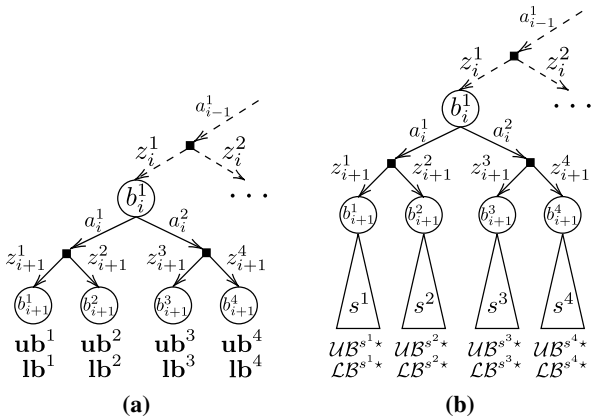


Fig. 3: (a) Leaf nodes are bounded using (5). **(b)** Adaptive Simplification, Subtrees are bounded using (10) and Alg. 2.

(see illustration in Fig. 2b). We discuss how we overcome this difficulty in the next section.

B. Adaptive Simplification

We address possible bounds overlapping by extending the definition of simplification as we envision it to be an *adaptive paradigm*. We denote *level of simplification* as how 'aggressive' the suggested simplification is. Consider the belief is represented by a set of samples (particles), as we do in Section V-C.1. Taking a small subset of particles to represent the simplified belief corresponds to *coarse* simplification. Taking many of them will correspond to *fine* simplification. Naturally, with this setting, we can define many discrete levels. We denote subscript i for s_i as the simplification level, where s_0 and s_n correspond, respectively, to coarsest and finest simplification levels. Additionally we denote superscript j for s^j as the index corresponding to the belief's tree index. E.g. in Fig. 1 for tree node b^4_{k+1} the corresponding simplification index is s^4 and it may assume any value of simplification: $s^4 \in \{s_0, s_1, ...s_n\}$.

Further, we assume bounds monotonically become tighter as the simplification level is increased and that the bounds for the finest simplification level s_n converge to the original problem. More formally denote $\overline{\Delta}^s(b,a) \triangleq \mathbf{ub}(b^s_i, b_i, a) - r(b_i, a)$ and $\underline{\Delta}^s(b,a) \triangleq r(b_i,a) - \mathbf{lb}(b^s_i, b_i, a)$.

Assumption 1. $\forall s \in [0, n-1]$ we get: $\overline{\Delta}^s(b,a) \geq \overline{\Delta}^{s+1}(b,a)$ and $\underline{\Delta}^s(b,a) \geq \underline{\Delta}^{s+1}(b,a)$.

Assumption 2. $\forall b_i, a$ we get: $\mathbf{ub}(b^{s_n}, b_i, a) = \mathbf{lb}(b^{s_n}, b_i, a) = r(b_i, a)$.

In the sequel we provide bounds that indeed satisfy these assumptions. A key question is how can we decide the appropriate level of simplification beforehand? We would like the coarsest level s_i that will enable eliminating actions/branches, i.e. lead us to Fig. 2c and not Fig. 2b. In Alg. 1 our adaptive simplification approach is summarized. The general idea is to break down recursively a given belief tree $\mathbb{T}$ into its sub-problems (subtrees), denoted as $\{\mathbb{T}_m\}_{m=1}^{|\mathcal{A}|}$, and solve each sub-problem with its simplification level s_i. Ultimately this would lead to the solution of the entire problem via (8). A potential computational issue is that increasing the simplification level repeatedly might be not worth it, since overall time for all levels calculations is suppressing the time it takes to solve the original problem. Fortunately, this is not an issue with our adaptive simplification, as discussed next.

Our adaptive simplification approach is based on two key observations. The *first key observation* is we can compare bounds from *different* levels of simplification when pruning. Our *second key observation* is that we can re-use calculations between different simplification levels, and thus avoid re-calculating simplification from scratch. In the following sections, we elaborate on each of these crucial aspects.

1) Comparing Bounds with Different Simplification Levels: Consider again some belief node b_i in the belief tree, and assume recursively for *each* of its children belief nodes b_{i+1} we already calculated the optimal policy $\pi^{\star}_{(i+1)+}(b_{i+1})$ and the corresponding upper and lower bounds $\mathcal{UB}^{s^l\star}(b_{i+1})$ and $\mathcal{LB}^{s^l\star}(b_{i+1})$, where s^l indicates the simplification level, and l corresponds to the belief tree nodes indexing notation. In general, the bounds for each belief node b_{i+1} can correspond to different simplification levels, as illustrated in Fig. 3b.

We now discuss how the simplification level is updated recursively, and revisit the process to calculate the optimal policy and the corresponding bounds for belief node b_i, previously described by Eqs. (8) and (9). Incorporating adaptive simplification, Eq. (8) is modified to

$$\begin{aligned}\mathcal{UB}^{s^j}(b_i, \{a^j_i, \pi^{\star}_{(i+1)+}\}) &= \mathbf{ub}(b^s_i, b_i, a^j_i) + \frac{1}{n_z}\sum_l \mathcal{UB}^{s^l\star}(b^l_{i+1}) \\ \mathcal{LB}^{s^j}(b_i, \{a^j_i, \pi^{\star}_{(i+1)+}\}) &= \mathbf{lb}(b^s_i, b_i, a^j_i) + \frac{1}{n_z}\sum_l \mathcal{LB}^{s^l\star}(b^l_{i+1}).\end{aligned} \tag{10}$$

Note this equation applies for each $a^j_i \in \mathcal{A}$, and as mentioned, each belief node b^l_{i+1} (one for each observation z^l_{i+1}) has its own simplification level s^l. In other words, for each b^l_{i+1}, s^l is the simplification level that was sufficient for calculating the bounds $\{\mathcal{UB}^{s^l\star}(b^l_{i+1}), \mathcal{LB}^{s^l\star}(b^l_{i+1})\}$ and the corresponding optimal policy $\pi^{\star}_{(i+1)+}(b^l_{i+1})$. Thus, when addressing belief node b_i in (10), for each belief node b^l_{i+1} and its corresponding simplification level s^l, these bounds are already available. Yet, we may still need to adapt the simplification level as we further discuss in Section V-B.2.

Further, as seen in (10), the immediate reward and the corresponding bounds $\mathbf{ub}$ and $\mathbf{lb}$, in general, can be calculated with their own simplification level s. In particular, when starting calculations, s could correspond to a default coarse simplification level, e.g. coarsest level s_0.

To define simplification level s^j of the bounds (10) we remind the belief tree is a discrete approximation to the expectation taken w.r.t future observations $z_i, i \in \{k, k+L\}$. We account for some number n_z of observations made in tree nodes (e.g. in Fig. 3b, $n_z = 2$)

$$s^j \triangleq \min\{s, s^{l_1}, s^{l_2}, ...s^{l_{n_z}}\}, \tag{11}$$

where $\{s^{l_1}, s^{l_2}, ...s^{l_{n_z}}\}$ represents the (generally different) simplification levels of belief nodes b^l_{i+1} considered in the

expectation approximation in (10). We explain the reason to define s^j as such in Section V-B.2.

As earlier, we wish to decide which action $a_i^\star \in \mathcal{A}$ is optimal from belief node b_i; the corresponding optimal policy would then be $\pi_{i+}^\star = \{a_i^\star, \pi_{(i+1)+}^\star\}$, where $\pi_{(i+1)+}^\star$ is the already-calculated optimal policy for belief node b_{i+1}^l that $a_i^\star$ leads to. See illustration in Fig. 3b.

Determining $a_i^\star$ requires eliminating all other candidate actions $a^j \in \mathcal{A}$, which involves comparing their corresponding bounds (10). Importantly, the bounds are analytical, i.e. they are *valid for all simplification levels*.

As earlier, we can compare bounds for different candidate actions and if the bounds do not overlap, perform pruning. For example, if for $a_i^1, a_i^2 \in \mathcal{A}$,

$$\mathcal{UB}^{s^1}(b_i, \{a_i^1, \pi_{(i+1)+}^\star\}) < \mathcal{LB}^{s^2}(b_i, \{a_i^2, \pi_{(i+1)+}^\star\}), \quad (12)$$

we can prune the a_i^1 branch. If the bounds are sufficiently tight and all branches but one were pruned, then the remaining action, in this case a_i^2, is announced as $a_i^\star$, and s^2 is announced as $s^\star$. Thus the above-mentioned optimal policy $\pi_{i+}^\star$ is constructed.

We now recall b_i itself has an index in the belief tree, with respect to the previous level. We denote it as b_i^l, considering the father node of b_i is b_{i-1}, and the lth corresponding observation z_i^l. At this point we get

$$\mathcal{UB}^{s^l\star}(b_i^l) = \mathcal{UB}^{s^\star}(b_i^l, \{a^\star, \pi_{(i+1)+}^\star\}), \quad (13)$$

and the same for $\mathcal{LB}^{s^l\star}(b_i^l)$. As in (9), (13) leads to bounds over the objective function

$$\mathcal{LB}^{s^l\star}(b_i^l) \leq J^\star(b_i^l) \leq \mathcal{UB}^{s^l\star}(b_i^l), \quad (14)$$

where b_i^l corresponds to the same notation as in (10), recursively. In general, pruning as done in (12) will not always hold, and we need to adapt level of simplification. We discuss in the next section how we do this, including re-use of calculations.

2) Adapting Simplification Level with Calculation Re-Use: For some belief node b_i in the belief tree, consider the bounds $\mathcal{UB}^{s^j}(b_i, \{a_i^j, \pi_{(i+1)+}^\star\})$ and $\mathcal{LB}^{s^j}(b_i, \{a_i^j, \pi_{(i+1)+}^\star\})$ from (10) for different actions $a_i^j \in \mathcal{A}$, that partially overlap and therefore could not be pruned. Each such action a_i^j can generally have its own simplification level s^j. We now iteratively increase the simplification level by 1. This can be done for each of the branches, if s^j is identical for all branches, or only for the branch with the coarsest simplification level. Consider now any such branch whose simplification level needs to be adapted from s^j to $s^j + 1$. Recall, that at this point, the mentioned bounds were already calculated, thus their ingredients, in terms of $\mathbf{ub}(b_i^s, b_i, a_i^j)$, $\mathbf{lb}(b_i^s, b_i, a_i^j)$ and $\{\mathcal{UB}^{s^l\star}(b_{i+1}), \mathcal{LB}^{s^l\star}(b_{i+1})\}_{l=1}^{n_z}$, involved in approximating the expectation in (10), are available. Recall also $s^j \triangleq \min\{s, s^{l_1}, s^{l_2}, ...s^{l_{n_z}}\}$ from (11), i.e. each element in $\{s, s^{l_1}, s^{l_2}, ...s^{l_{n_z}}\}$ is either equal or larger than s^j. We now discuss both cases, starting from the latter.

As we assumed bounds to improve monotonically as simplification level increases, see Assump. 1, for any $s^l > s^j + 1$ we already have readily available bounds $\{\mathcal{UB}^{s^l\star}(b_{i+1}), \mathcal{LB}^{s^l\star}(b_{i+1})\}$ which are tighter than those that would be obtained for simplification level $s^j + 1$. Thus, we can *safely skip* the calculation of the latter and use the existing bounds from level s^l as is.

For the former case, i.e. $s^l = s^j$, we now have to adapt the simplification level to $s^j + 1$ by calculating the bounds $\{\mathcal{UB}^{(s^l+1)\star}(b_{i+1}), \mathcal{LB}^{(s^l+1)\star}(b_{i+1})\}$. Here, our *key insight* is that, instead of calculating these bounds from scratch, we can re-use calculations between different simplification levels, in this case, from level s^l. As the bounds from that level are available, we can identify only the incremental part that is "missing" to get from simplification level s^l to $s^l + 1$, and update *analytically* the existing bounds $\{\mathcal{UB}^{s^l\star}(b_{i+1}), \mathcal{LB}^{s^l\star}(b_{i+1})\}$ to recover $\{\mathcal{UB}^{(s^l+1)\star}(b_{i+1}), \mathcal{LB}^{(s^l+1)\star}(b_{i+1})\}$ exactly. The same argument applies also for bounds over momentary rewards. In Section V-D.3 we apply this approach to a specific simplification and reward function.

We can repeat iteratively the above process of increasing the simplification level until we can prune all branches but one. This means each subtree will be solved maximum once, per simplification level. Since we assumed the simplification converges to the original problem for the finest level s_n, see Assump. 2, we are guaranteed to eventually disqualify all sub-optimal branches. Moreover, due to the discussed-above calculation re-use, in the worst case, adapting the simplification all the way up to the finest level s_n, is roughly equivalent to solving the original problem. We address this aspect explicitly in Section V-D.3. For a detailed illustrative example w.r.t. Fig. 3b see Sec. III in [19].

Algorithm 1 Simplified Information Theoretic Belief Space Planning

procedure FIND OPTIMAL POLICY(belief-tree: $\mathbb{T}$)
  $s \leftarrow s_0$
  return ADAPT SIMPLIFICATION($\mathbb{T}$,s)
end procedure
procedure ADAPT SIMPLIFICATION(belief-tree: $\mathbb{T}$, s_i)
  if $\mathbb{T}$ is a leaf **then**
    return $\{\mathbf{lb}, \mathbf{ub}\}$ ▷ Corresponds to immediate reward bounds (4).
  end if
  Set simplification level: $s \leftarrow s_i$
  for all subtrees $\mathbb{T}'$ in $\mathbb{T}$ **do**
    ADAPT SIMPLIFICATION($\mathbb{T}'$,s)
    Calculate $\mathcal{LB}^{s^j}, \mathcal{UB}^{s^j}$ according to s and (10)
  end for
  Using $\{\mathcal{LB}^{s^j}, \mathcal{UB}^{s^j}\}_{j=1}^{|\mathcal{A}|}$ and Alg. 2 prune branches
  while not all $\mathbb{T}'$ but 1 in $\mathbb{T}$ pruned **do**
    Increase simplification level: $s \leftarrow s + 1$
    ADAPT SIMPLIFICATION($\mathbb{T}$,s)
  end while
  Update $\{\mathcal{LB}^{s^j\star}, \mathcal{UB}^{s^j\star}\}$ according to (13)
  return optimal action branch that left $a^\star$ and $\{\mathcal{LB}^{s^j\star}, \mathcal{UB}^{s^j\star}\}$.
end procedure

In the following sections, we present a specific simplification along with derivations that show it holds the mentioned mathematical properties. Our described approach is summarized in Algs. 1 and 2.

Algorithm 2 Prune Branches

procedure PRUNE
  Input: (belief-tree root, b; bounds of root's children, $\{\mathcal{LB}^m, \mathcal{UB}^m\}_{m=1}^{C}$)
  ▷ C is the number of child branches going out of b.
  $\mathcal{LB}^\star \leftarrow \max_m \{\mathcal{LB}^m\}_{m=1}^{C}$
  for all children of b **do**
    if $\mathcal{LB}^\star > \mathcal{UB}^m$ **then**
      prune child m from the belief tree
    end if
  end for
end procedure

C. Bounds

We now delve deeper to the bounds our chosen simplification suggests.

1) Belief Representation and Chosen Simplification: As mentioned, we use Particle Filter for belief update. This means the belief is represented as a set of weighted particles,

$$b \triangleq \{x^i, w^i\}_{i=1}^{N}, \tag{15}$$

where N is a tune-able parameter specifying the desired number of particles.

Suggested Simplification: Given the belief representation (15), the simplified belief is a subset of N^s particles, sampled from the original belief, where $N^s \leq N$. More formally:

$$b_k^s \triangleq \{(x^i, w^i) \mid i \in A_k^s, A_k^s \subseteq \{1, 2, \ldots, N\}, |A_k^s| = N_k^s\}, \tag{16}$$

where A_k^s is the set of particle indices comprising the simplified belief b_k^s for time k.

Increasing the level of simplification is done *incrementally*. Specifically, consider $|A^s| = N^s$ and to get to the next simplification level we add m particles with indices $j \in B,\ |B| = m$. Then the following holds: $A^s \cap B = \emptyset,\ A^{s+1} = A^s \cup B,\ N^{s+1} = N^s + m$.

2) Bounding the Differential Entropy: We consider a common reward function, the differential entropy. As one of our key contributions, in this section we derive novel analytical upper and lower bounds $\mathbf{lb}$ and $\mathbf{ub}$ considering this reward function, assuming a sampling-based belief representation (15) and the corresponding simplified belief (16). These bounds can then be used within our general simplification framework presented in Sections V-A and V-B.

Under this setting approximating differential entropy of the belief is not an easy task. To calculate $\mathcal{H}(b[x_k]) = -\int b[x_k] \cdot \log(b[x_k])\, dx_k$, one must have access to the manifold representing the belief. Several approaches exist. One of them is using Kernel Density Estimation as done, e.g., by [7]. Here, we consider the method proposed by [20]:

$$\hat{\mathcal{H}}(b_{k+1}) \triangleq \underbrace{\log\left[\sum_i \mathbb{P}(z_{k+1} \mid x_{k+1}^i) w_k^i\right]}_{\text{(a)}} \tag{17}$$

$$\underbrace{-\sum_i w_{k+1}^i \cdot \log\left[\mathbb{P}(z_{k+1} \mid x_{k+1}^i) \sum_j \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j\right]}_{\text{(b)}},$$

where i indexes particles and their corresponding weight as in (15) and (16).One can observe this method requires access to samples representing both b_k and b_{k+1}; thus, this corresponds to an information-theoretic reward of the form $r(b_k, b_{k+1})$. Utilizing the chosen simplification (16) we can now upper and lower bound 17. We do so below by bounding term (b) which is the source of complexity. First, as the models are known, we define $m \triangleq \max\{\mathbb{P}(x_{k+1} \mid x_k, a_k)\}$, the max value the given motion model can assume.

Theorem 1. *Term* (b) *in* (17) *can be upper and lower bounded via simplification as*

$$(b) \geq -\sum_{i \in \neg A_{k+1}^s} w_{k+1}^i \cdot \log\left[m \cdot \mathbb{P}(z_{k+1} \mid x_{k+1}^i)\right]$$

$$-\sum_{i \in A_{k+1}^s} w_{k+1}^i \cdot \log\left[\mathbb{P}(z_{k+1} \mid x_{k+1}^i) \sum_j \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j\right]$$

$$(b) \leq -\sum_i w_{k+1}^i \cdot \log\left[\sum_{j \in A_k^s} \mathbb{P}(z_{k+1} \mid x_{k+1}^i) \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j\right]$$

See proof in Supplementary [19] Sec. I.

Finally, bounding (17) using Theorem 1 corresponds, in our general framework from Sections V-A and V-B, to (5). Note we consider simplified and non-simplified belief nodes pairs for the bounds calculations which still settles with (5).

D. Bounds Analysis

1) Convergence: We now analyze convergence of the simplification described in Section V-C.1. Since simplifying to the level of s_n means the simplified belief is just the original belief, we get: $N^s = N \Rightarrow b = b^s \Rightarrow r(b^s, a) = r(b, a)$. Furthermore, if $N^s = N$ then $\neg A^s = \emptyset$ and the first term of the lower bound is zero. Further, $A^s = \{1, 2, ..., N\}$ and thus the upper and lower bounds in Theorem 1 coincide and equal to term (b) from (17). Meaning, under our chosen simplification, the bounds converge to the original rewards for the non-simplified original belief. Where we consider as lower bounds term (a) from (17) with lower bound to term (b), and similarly we build the upper bounds

2) Complexity Analysis: The original formulation of [20] suggests complexity of $O(N^2)$ where N is as in (15). The derivations in Section V-C.2 suggest complexity of $O(N^s \cdot N)$, where N^s is the maximal number of particles needed for pruning, for some belief node. Altogether, time saved for all belief nodes in the tree will result in the total speedup of our approach.

3) Re-use of Calculations: The unique structure of the simplification allows us to cache the calculation from a previous simplification level. Specifically, moving from simplification level s to level $s+1$, corresponds to adding some m additional particles to b^s in order to get b^{s+1}. For the lower bound in Theorem 1 we can just cache the final result and augment the sums with the calculations corresponding to the m missing particles. For the upper term in Theorem 1, we need to cache the inner sums of the $\log$, augment them with the missing m particles calculations, and re-weight it using the outer sum weights. Note space complexity remains the same since we are already caching the particles and weights for each belief. This re-use of calculations results in time complexity, going up from simplification level s_0 to level s_n

being the same order as solving the original problem in the first place. Thus, making simplification is worthwhile always (up to some constant overhead).

VI. Experimental Setting and Results

We consider an autonomous navigation to goal scenario with continuous state and observation spaces. First, we experiment with a passive case, i.e. a given policy, to demonstrate our novel entropy bounds behavior. Second, we show how the bounds can use us to speed up POMDP planning as explained in Section V. All experiments were conducted on a laptop with Intel i7-9850H CPU 2.59GHz with 16 GB RAM.

A. Experimental Setting - 2D Continuous Light-Dark

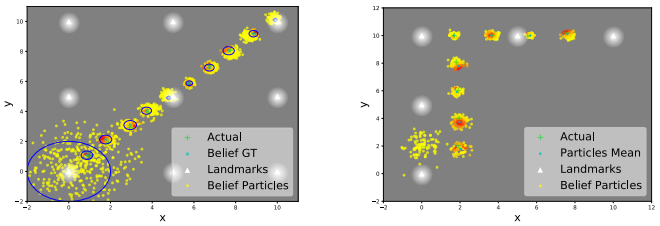
We consider a *2D continuous Light-Dark problem*. The robot starts at some unknown point $x_0 \in \mathbb{R}^2$. In this world, there are spatially scattered beacons with known locations. Near the beacons, the attained observations are less 'noisy'. The goal is to get to the goal $x^t \in \mathbb{R}^2$ (upper right corner of the world). Initial belief is $b[x_0] = \mathcal{N}(x_0, I \cdot \sigma_0)$, motion and observation models are $T = \mathbb{P}(x' \mid x, a) = \mathcal{N}(x + a, I \cdot \sigma_T)$, $\mathcal{O} = \mathbb{P}(z \mid x) = \mathcal{N}(x - x^b, I \cdot \sigma_{\mathcal{O}} \cdot \max\{r, r_{min}\})$ respectively, where r is the robot's distance to the nearest beacon whose known location is x^b, and r_{min} is a tune-able parameter. For all experiments, the belief is approximated by a set of N weighted samples as in (15). This setting implies the belief at each time step is Gaussian and can be inferred exactly using Kalman Filter (KF). Thus, the differential entropy has a closed-form and can be calculated across the simulation. Note we consider this Gaussian case only as a reference point and our approach is applicable to any distribution.

B. Differential Entropy Approximations

To verify our bounds behavior we consider a passive scenario over the setting described in VI-A. The robot moves diagonally to the goal (Fig. 4a). Along the way, it passes close by two beacons. Consequentially, the robot's uncertainty decreases. In Fig 5a we plot the bounds along with the original approximation (17), a KDE approximation (as done by [7]), the actual differential entropy and naive approximation using discrete entropy of the particles weights: $h(b) = -\sum_i w^i \cdot \log w^i$. We experiment with a changing number of particles and it is clear the bounds converge to the original reward (differential entropy approximation) (17).

C. Planning in 2D environment

We demonstrate the simplification speedup when planning in a continuous 2D scenario. The setting is as in Section VI-A. However, now the robot is given the ability to plan a parametric number of L steps into the future. After the planning session is done, the robot executes the first action out of the calculated optimal policy, acquires a new observation, updates the belief, and performs planning again (and so on). The reward function is $-r(b,a) = \mathop{\mathbb{E}}_{x\sim b}\{\|x - x^t\|_1\} + \hat{\mathcal{H}}(b)$, where the first term is the expected distance to goal, and $\hat{\mathcal{H}}(b_k)$ is the approximation of the differential entropy (17).



(a) **(b)**

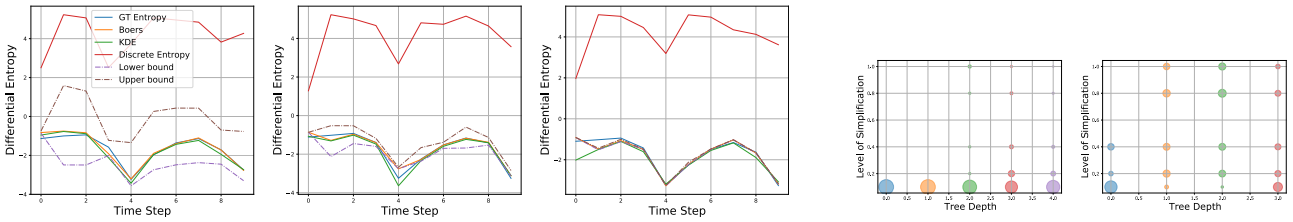
Fig. 4: **(a)** Localization with a predefined policy in an environment represented by known landmarks. Particles are colored according to weight (high-red, small-yellow). Blue ellipses correspond to estimated uncertainty covariance by a KF. **(b)** Planning simulation, setting II. At each time step the robot is planning L step into the future and then executes the first action from the calculated optimal policy. The plot shows belief evolution in terms of particles sets along the actual trajectory taken by the robot.

We apply our approach considering different tree structures. Specifically, we evaluate our approach on a POMCP-like tree (deep and sparse) [2], DESPOT-like sparse tree ([15]), and on a shallow and 'thick' tree like the one generated by [22]. We provide a full explanation of how these trees are built in the caption of Table I. The reported results in Table I are the mean planning time (in seconds) of our approach, compared to calculating the objective using original rewards. We experiment with a changing number of particles and planning horizon.

We consider two settings: 'I' and 'II'. Setting 'II', illustrated in Fig. 4b, is more complex as the robot needs to move from the bottom left corner to the top right and the action space is {left, right, up, down}. Setting 'I' is easier: The robot needs to move from an initial point x_0 to the same height point x_L, which means the optimal path is just a straight line, while the action space is {left, right}. This easy setting allows the robot to utilize simplification to its full extent. Empty cells in Table I correspond to runs that planning session (for a regular objective) took longer than 35 [sec] and was stopped. Initial simplification level was set to $s_0 = 0.1$, i.e., $N^{s_0} = 0.1 \cdot N$ and specifically the levels are $s_i \in \{0.1, 0.2, 0.4, 0.8, 1.0\}$.

It is clear from Table I, using our suggested simplification is a favorable approach, leading to speedup in all of the conducted experiments when compared to the Sparse Sampling baseline. Since the simplification bounds are analytical and used for eliminating branches in the belief tree of the original problem, the *same optimal action* is obtained with or without our simplification. In other words, we demonstrate a significant speedup while obtaining the same solution.

In Fig. 5 we can get a glimpse into how our adaptive simplification performing in the tree depth for Settings I and II. The shown plots are histograms that tell us what are the levels of simplification in the tree needed for pruning. It can be seen that indeed for Setting I the simplification is performing extremely well thus saving a lot of time 5b. In the more difficult Setting II, indeed higher simplification levels are more common 5c. Nevertheless, as seen in Table I, we still get a significant speedup, while the speedup for Setting I is even more drastic. This implies our simplification can identify by itself situations where we can save resources (computation time) and all this without compromising on the accuracy of the desired solution.



(a) **(b)** **(c)**

Fig. 5: **(a)** Differential entropy approximations and bounds. Calculations were done using $N = 200$ particles. From left to right: Simplification is $N^s = \{0.1, 0.5, 0.9\} \cdot N$. **(b)** and **(c)** show simplification level histograms vs. tree depth for entire simulation (10 planning sessions). **(b)** Setting I using $N = 50$ and Horizon of 4. **(c)** Setting II using $N = 20$ and Horizon of 3. x-axis corresponds to tree nodes of some depth in the belief tree. y-axis corresponds to the simplification level needed to achieve pruning. The scale of the circles corresponds to how many nodes were in that category. Circle scales are normalized since the number of nodes grows exponentially going down the tree.

TABLE I: Mean time per planning session. Each table entry is (*original objective time/simplified approach time*). Results are in Seconds. The second row indicates the number of particles used. DESPOT-like belief tree [15] was built by expanding all actions in every level of the tree but only making a single observation in each level, i.e. $n_z = 1$. POWSS-like tree [22] was built by expending each belief node for all actions, and generating an observation for each particle of the belief, i.e. number of observations when branching is as the number of particles. POMCP-like tree [2] was built using five 'rollout's starting from the root of the tree. In each rollout down the tree, we randomly choose if to expand a new node by taking an action that was not taken previously from that node or to go down the tree using nodes that were already expanded from previous rollouts.

Simulation	Horizon	[15] Tree			Horizon	[22] Tree			Horizon	[2] Tree		
		20	50	100		10	20	30		20	50	100
Setting I	1	0.124/**0.043**	0.741/**0.192**	2.892/**0.667**	1	0.554/**0.287**	4.065/**1.437**	12.908/**3.953**	5	1.13/**0.776**	6.625/**2.008**	28.19/**7.232**
	2	0.364/**0.129**	2.196/**0.584**	8.616/**2.042**	2	11.02/**5.386**	-	-	10	2.648/**2.555**	15.342/**8.214**	-
	3	0.853/**0.339**	5.059/**1.324**	19.899/**4.658**	3	-	-	-	15	4.2/**3.677**	26.205/**20.174**	-
Setting II	1	0.245/**0.099**	1.513/**0.4**	5.855/**2.018**	1	1.112/**0.953**	8.501/**5.143**	26.375/**11.977**	5	1.383/**0.733**	8.417/**3.864**	33.244/**10.97**
	2	1.209/**0.738**	7.195/**3.821**	30.638/**13.49**	2	-	-	-	10	2.985/**2.112**	17.293/**6.092**	-
	3	5.027/**3.212**	31.515/**18.288**	-	3	-	-	-	15	4.53/**3.701**	27.712/**11.385**	-

VII. Conclusion

In this paper we have introduced SITH-BSP, an algorithmic paradigm able to speedup calculations when performing online POMDP planning, considering general distributions and belief-dependent rewards. The lion part of our approach, definition of simplification over sparse sampling planning, is mathematically formulated in a general manner. It can be easily extended to many directions and may prove to be applicable for existing or future approaches. The more specific aspects of our approach (assuming Particle Filter for belief approximation) provides novel derivations for bounds over the differential entropy approximation and may be taken to other areas in the vast field of robotics. We have shown how the approach assumes an adaptive form, while re-using calculations and thus gives the exact optimal solution to the original problem in an effective manner.

References

[1] C. Papadimitriou and J. Tsitsiklis, "The complexity of markov decision processes," *Mathematics of operations research*, vol. 12, no. 3, pp. 441–450, 1987.

[2] D. Silver and J. Veness, "Monte-carlo planning in large pomdps," in *Advances in Neural Information Processing Systems (NIPS)*, 2010, pp. 2164–2172.

[3] T. Smith and R. Simmons, "Heuristic search value iteration for pomdps," in *Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2004, pp. 520–527.

[4] Z. Sunberg and M. Kochenderfer, "Online algorithms for pomdps with continuous state, action, and observation spaces," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 28, no. 1, 2018.

[5] N. P. Garg, D. Hsu, and W. S. Lee, "Despot-α: Online pomdp planning with large state and observation spaces," in *Robotics: Science and Systems (RSS)*, 2019.

[6] M. H. Lim, C. J. Tomlin, and Z. N. Sunberg, "Voronoi progressive widening: Efficient online solvers for continuous space mdps and pomdps with provably optimal components," *arXiv preprint arXiv:2012.10140*, 2020.

[7] J. Fischer and O. S. Tas, "Information particle filter tree: An online algorithm for pomdps with belief-based rewards on continuous domains," in *Intl. Conf. on Machine Learning (ICML)*, Vienna, Austria, 2020.

[8] V. Thomas, G. Hutin, and O. Buffet, "Monte carlo information-oriented planning," *arXiv preprint arXiv:2103.11345*, 2021.

[9] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using Rao-Blackwellized particle filters," in *Robotics: Science and Systems (RSS)*, 2005, pp. 65–72.

[10] A. Singh, A. Krause, C. Guestrin, and W. Kaiser, "Efficient informative sensing using multiple robots," *J. of Artificial Intelligence Research*, vol. 34, pp. 707–755, 2009.

[11] M. Kearns, Y. Mansour, and A. Y. Ng, "A sparse sampling algorithm for near-optimal planning in large markov decision processes," *Machine learning*, vol. 49, no. 2, pp. 193–208, 2002.

[12] M. Hoerger, H. Kurniawati, and A. Elfes, "Multilevel monte-carlo for solving pomdps online," in *Proc. International Symposium on Robotics Research (ISRR)*, 2019.

[13] L. Kocsis and C. Szepesvári, "Bandit based monte-carlo planning," in *European conference on machine learning*. Springer, 2006, pp. 282–293.

[14] M. Kochenderfer, T. Wheeler, and K. Wray, *Algorithms for Decision Making*. MIT Press, 2022.

[15] N. Ye, A. Somani, D. Hsu, and W. S. Lee, "Despot: Online pomdp planning with regularization," *JAIR*, vol. 58, pp. 231–266, 2017.

[16] N. Roy, G. J. Gordon, and S. Thrun, "Finding approximate pomdp solutions through belief compression," *J. Artif. Intell. Res.(JAIR)*, vol. 23, pp. 1–40, 2005.

[17] K. Elimelech and V. Indelman, "Simplified decision making in the belief space using belief sparsification," *Intl. J. of Robotics Research*, 2021, accepted.

[18] M. Shienman, A. Kitanov, and V. Indelman, "Ft-bsp: Focused topological belief space planning," *IEEE Robotics and Automation Letters (RA-L)*, vol. 6, no. 3, pp. 4744–4751, July 2021.

[19] O. Sztyglic and V. Indelman, "Speeding up pomdp planning via simplification - supplementary material," Technion - Israel Institute of Technology, Tech. Rep., 2022. [Online]. Available: https://tinyurl.com/3h945vrk

[20] Y. Boers, H. Driessen, A. Bagchi, and P. Mandal, "Particle filter based entropy," in *2010 13th International Conference on Information Fusion*, 2010, pp. 1–8.

[21] O. Sztyglic, A. Zhitnikov, and V. Indelman, "Simplified belief-dependent reward mcts planning with guaranteed tree consistency," Technion - Israel Institute of Technology, Tech. Rep., 2021.

[22] M. H. Lim, C. Tomlin, and Z. N. Sunberg, "Sparse tree search optimality guarantees in pomdps with continuous observation spaces," in *Intl. Joint Conf. on AI (IJCAI)*, 7 2020, pp. 4135–4142.