

Planning with SiMBA: Motion Planning under Uncertainty for Temporal Goals using Simplified Belief Guides

Qi Heng Ho, Zachary N. Sunberg, and Morteza Lahijanian

Abstract— This paper presents a new multi-layered algorithm for motion planning under motion and sensing uncertainties for Linear Temporal Logic specifications. We propose a technique to guide a sampling-based search tree in the combined task and belief space using trajectories from a simplified model of the system, to make the problem computationally tractable. Our method eliminates the need to construct fine and accurate finite abstractions. We prove correctness and probabilistic completeness of our algorithm, and illustrate the benefits of our approach on several case studies. Our results show that guidance with a simplified belief space model allows for significant speed-up in planning for complex specifications.

I. INTRODUCTION

As robots become more advanced, the expectation for them to perform tasks with higher complexities increases. It is thus an essential challenge to enable efficient planning for complex requirements. In addition, real-world robots must be able to reason about both motion and sensor uncertainty while executing such tasks. For instance, an autonomous underwater rover often has noisy motion due to water currents and gets accurate GPS measurements only at the surface; under the surface sensor readings are highly noisy. The combination of accounting for these uncertainties with the need to provide guarantees for task completion makes the planning problem very difficult. This paper focuses on this challenge and aims to develop an efficient framework for planning under uncertainty with complex specifications.

Linear Temporal Logic (LTL) is a principled formalism for expressing complex temporal tasks for robotic systems [1]–[3]. For instance, a robot tasked with “visit A and B in any order, and then go to C while avoiding D” can be expressed precisely as an LTL formula. The primary existing LTL motion planning frameworks for continuous state and action spaces are designed for deterministic systems [4]–[7].

To address LTL synthesis problems for systems with uncertainty, works [8]–[10] focus solely on motion uncertainty. They first abstract the evolution of the robot in the environment into a finite Markov Decision Process (MDP), and then synthesize a policy that maximizes the probability of successfully completing the LTL specification. For observation uncertainty, LTL synthesis on Partially Observable MDPs (POMDPs) have recently been proposed for moderately-sized discrete space problems [11], or through finite state abstraction of continuous spaces [12]. However, a common limitation of these works is the need for a finite abstraction from continuous states and actions to discrete ones. There

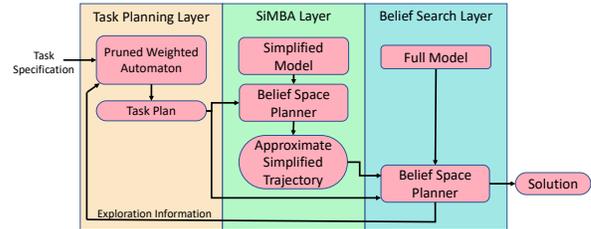


Fig. 1: Planning with SiMBA Framework

are currently no methods that can give practically useful guarantees for general continuous-space POMDPs.

For continuous spaces, recent works in motion planning under uncertainty combine the power of sampling-based methods [13] with point-wise chance-constraints on the probability of safety [14]–[20]. This is a relaxation of maximizing success probability, but it allows for fast and scalable motion planning with robustness guarantees. Most works focus on motion uncertainty [14], [15], [21], but recent works [17]–[20] introduce extensions to account for sensor uncertainty. While they provide efficient planning under uncertainty, they are limited to simple task of A-to-B planning. Recent work [22] extends chance constrained planning to LTL tasks and minimizes the probability of task failure, but their approach is limited to systems under motion uncertainty.

A main challenge in dealing with both uncertainty and complex specifications is the need to simultaneously account for both the belief and task spaces. Doing so results in extremely large search spaces. Principled guidance of a low-level motion tree has been shown to improve tractability in the deterministic setting for both single and multi-temporal goals planning, using layers of planning [4], [5], [23], [24] or heuristic guidance [6]. For stochastic systems, [25] proposes a heuristic to guide tree extension, but their framework is only designed for dynamics noise. Extending these techniques to settings with both motion and measurement uncertainty is non-trivial because of the difficulty in constructing a guidance mechanism that captures belief dynamics well.

In this paper, we present a multi-layered framework to synthesize motion plans for LTL tasks for systems under both motion and measurement uncertainty. The LTL tasks are defined in the belief space with a user-defined robustness requirement. We show that our framework provides guarantees on task completion and probabilistic completeness.

Our approach has two novel characteristics: First, we introduce a method to plan over an automaton that represents the LTL task directly that avoids the need for fine abstraction, by pruning infeasible edges. Second, in order to make the problem computationally tractable, we propose to use a sim-

Authors are with the department of Aerospace Engineering Sciences at the University of Colorado Boulder, CO, USA {*firstname.lastname*}@colorado.edu

plified model that accounts for both motion and measurement uncertainty. We refer to this notion as Simplified Models with Belief Approximation (SiMBA), which we use to rapidly find a continuous path that heuristically guides the search for a satisfying motion plan for the full system. Our evaluation shows that the addition of SiMBA for guidance improves solution time significantly. To the best of our knowledge, this is the first work that uses trajectories from a simplified model in the belief space to bias sampling-based search for stochastic systems under complex tasks.

In summary, the contributions of this paper are five-fold: (i) a framework for planning under LTL specifications for continuous systems under motion and measurement uncertainty with (ii) guarantees on task completion and probabilistic completeness; (iii) planning over a pruned automaton that alleviates the state explosion problem in abstraction-based methods; (iv) improving efficiency by using a simplified model in the belief space to provide continuous trajectory guides for belief planning, and (5) a series of illustrative case studies and benchmarks.

II. PROBLEM FORMULATION

Consider a robot with both motion and sensing uncertainty tasked with a complex navigation task in a bounded workspace $W \subset \mathbb{R}^d, d \in \{2, 3\}$. We are interested in computing a motion plan for the robot to achieve its task with guarantees. Below, we formalize this problem.

The robot has linear or linearizable motion and measurement models given by:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + \omega_k, & \omega_k &\sim \mathcal{N}(0, Q), \\ z_k &= Cx_k + Du_k + \nu_k(x_k), & \nu_k(x_k) &\sim \mathcal{N}(0, \mathcal{R}(x_k)), \end{aligned} \quad (1)$$

where $x_k \in X \subset \mathbb{R}^n$ is the state, $u_k \in U \subset \mathbb{R}^m$ is the input, and $z_k \in \mathbb{R}^p$ is the measurement with their corresponding matrices $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, and $C \in \mathbb{R}^{p \times n}$. Noise terms ω_k and ν_k are i.i.d. random variables with zero-mean Gaussian distributions and covariance matrices Q and $\mathcal{R}(x_k)$, respectively. Note that covariance $\mathcal{R}(x_k)$ depends on state x_k , i.e., it is state-varying measurement noise. This describes problems common in robotics where measurements may be available only in parts of the environment or measurement quality depends on the location (state) of the robot.

The evolution of the robotic system in (1) can be described by a discrete-time Gauss-Markov process [26]. The robot state x_k at each time step is a random variable with a Gaussian distribution, i.e., $x_k \sim b_k = \mathcal{N}(\hat{x}_k, \Sigma_k)$, where $b_k \in \mathcal{B}$ is referred to as the *belief* of the robot state, \mathcal{B} is the *belief space*, and \hat{x}_k and Σ_k are the state mean and covariance matrix, respectively. A *belief trajectory* $\mathbf{b} = b_0 b_1 b_2 \cdots b_f$ is a sequence of beliefs.

A. Linear Temporal Logic over Finite Traces for Belief Tasks

The robot is tasked with a temporal specification over the belief space, similar to [12], [27]–[30]. Let $\mathcal{P} = \{p_1, \dots, p_n\}$ be a set of convex polytopic regions in the state space X , i.e., $p_i \subset X$ for all $1 \leq i \leq n$. Further, let $\alpha_i \in [0, 1]$ be a probability threshold (chance constraint) on p_i . Then, we define atomic proposition $\pi_i^{\alpha_i}$ associated

with region p_i as follows: $\pi_i^{\alpha_i}$ is true for belief b_k iff $P(x_k \in p_i) > 1 - \alpha_i$. This definition states that a proposition is true if the probability that a belief state is in a region p_i is at least $1 - \alpha_i$. Hence, the set of atomic propositions is $\Pi = \{\pi_i^{\alpha_i} \mid i = 1, \dots, n\}$.

We define a labeling function $L : \mathcal{B} \rightarrow 2^\Pi$ that maps a belief $b \in \mathcal{B}$ to the set of atomic propositions in Π that are true for b . Then, a belief trajectory $\mathbf{b} = b_0 \cdots b_f$ generates an observation trace (word) $\sigma = \sigma_0 \cdots \sigma_f$ via the labeling function such that $\sigma_i = L(b_i)$ for all $0 \leq i \leq f$. Each $\sigma_i \in 2^\Pi$ is called a symbol.

Definition 1 (LTLf Syntax [1]). *Let Π be a set of atomic propositions. Then, an LTLf formula over Π is recursively defined by*

$$\phi := \pi^\alpha \mid \neg\phi \mid \phi \vee \phi \mid \mathcal{X}\phi \mid \phi \mathcal{U}\phi$$

where $\pi^\alpha \in \Pi$ is an atomic proposition, \neg (“negation”) and \vee (“or”) are Boolean operators, and \mathcal{X} (“next”) and \mathcal{U} (“until”) are the temporal operators.

From this definition, one can derive other standard temporal operators, e.g., the \diamond (“eventually”) and \square (“globally”) operators are given by $\diamond\phi \equiv \top \mathcal{U}\phi$ and $\square\phi \equiv \neg\diamond\neg\phi$.

LTLf formulas are interpreted over finite words in $(2^\Pi)^*$. The semantics of LTLf can be found in [1]. We say a finite trace σ satisfies formula ϕ , denoted by $\sigma \models \phi$, iff $\sigma, 0 \models \phi$. Additionally, we say that a belief trajectory satisfies a specification ϕ , written $\mathbf{b} \models \phi$, if its observation trace (word) $\sigma = L(b_0)L(b_1) \cdots L(b_n)$ satisfies ϕ , i.e., $\sigma \models \phi$.

Note that, for a small α , π_i^α requires being in region p_i with a high probability. The negation of it, $\neg\pi_i^\alpha = P(x_k \in p_i) \leq 1 - \alpha$, still allows being in region p_i with a relatively high probability for $\alpha \ll 1$. In many applications, however, one may be interested in expressing a task for being in a region with high probability in one part of a formula and then avoiding the same region also with high probability in another part of the formula. Similar to [31], we can do this by capturing the more intuitive converse of π_i^α by adding new atomic propositions. This can be done by associating to each p_i a new atomic proposition $\tilde{\pi}_i^\alpha$ that is true for belief b_k iff $P(x_k \notin p_i) \geq 1 - \alpha$, such that $\tilde{\pi}_i^\alpha$ represents $\neg\pi_i^{1-\alpha}$.

B. Planning Problem

We seek a motion plan that satisfies a given LTLf formula ϕ . Assuming that the robot is equipped with a feedback trajectory-following controller, a motion plan can then be characterized as a sequence of control inputs coupled with its corresponding nominal state trajectory. Let $\tilde{U}^{0,t} = (\tilde{u}_0, \tilde{u}_1, \dots, \tilde{u}_{t-1})$ be a sequence of control inputs. Then, given an initial belief b_0 , a nominal trajectory $\tilde{X}^{x_0, x_t} = (\tilde{x}_0, \tilde{x}_1, \dots, \tilde{x}_t)$ is obtained by applying $\tilde{U}^{0,t}$ to the nominal system dynamics $\tilde{x}_{k+1} = A\tilde{x}_k + B\tilde{u}_k$. This motion plan $(\tilde{U}^{0,t}, \tilde{X}^{x_0, x_t})$ is then executed online, given online state estimates \hat{x}_k , via the stabilizing, trajectory-following controller

$$u_k = \tilde{u}_{k-1} - K(\hat{x}_k - \tilde{x}_k), \quad (2)$$

where K is the feedback control gain.

Problem 1. Given System (1), a set of probabilistic atomic propositions $\Pi = \{\pi_1^{\alpha_1}, \dots, \pi_n^{\alpha_n}\}$ defined over regions \mathcal{P} , LTLf formula ϕ defined over Π , and closed-loop control gain K , find a motion plan (\tilde{U}, \tilde{X}) as a pair of sequence of nominal controls $\tilde{U} = (\tilde{u}_0, \dots, \tilde{u}_{k-1})$ for some $k \geq 1$ and its resulting nominal trajectory $\tilde{X} = (\tilde{x}_0, \dots, \tilde{x}_k)$ such that when executed via the controller in (2) the resulting belief trajectory \mathbf{b} satisfies the LTLf formula, i.e., $\mathbf{b} \models \phi$.

The main challenge in this problem is that the search space is extremely large. This is because planning must be performed in the composition of the belief space with the task space (representation of the set of all satisfying words). For an n -dimensional state space, the Gaussian belief space adds $O(n^2)$ dimensions, and for a task formula of size $|\phi|$, the size of the task can be as large as $2^{2^{|\phi|}}$ [1]. Hence, to make the problem computationally tractable, the search must be performed with informed guidance. Existing methods for guiding search in the state space through, e.g., discrete abstractions or geometric planning, fail to capture belief information, leading to uninformative guides. To address these challenges, we aim to design a framework that reduces dependence on fine, discrete abstractions, while efficiently searching in the belief space. Specifically, we seek to generate informed guidance via simplified models.

III. BELIEF SPACE PLANNING WITH SIMPLIFIED BELIEF APPROXIMATIONS

We present a modular framework to solve Problem 1. The framework is inspired by [5], [21], [23] and consists of three main layers: task planning layer, SiMBA guide layer, and belief search layer, as depicted in Fig. 1.

We first translate the LTLf formula ϕ into a minimal Deterministic Finite Automaton (DFA). Then, we prune unrealizable transitions of the DFA, and use the pruned DFA for task planning. At each iteration, we find the shortest path on the DFA that corresponds to a ϕ -satisfying trace to use as a task plan. Next, at the SiMBA guide layer, we use a simplified model of the system to rapidly find a path annotated with DFA states that obeys the task plan. We call this technique Simplified Models with Belief Approximation (SiMBA). Finally, at the belief search layer, a belief tree searches for a satisfying trajectory for the full system. The growth of this tree is biased to search around the hybrid annotated path while ensuring the motion plan satisfies ϕ .

A. Belief Propagation and Labeling

Here, we establish the relationship between the evolution of the distribution over robot state, atomic predicates, and labels. During offline planning, following [18], [19], given a stabilizing feedback controller in (2) and online state estimate \hat{x}_k , we parameterize the belief at each time step based on the closed-loop dynamics as

$$x_k \sim b_k = \mathcal{N}(\tilde{x}_k, \Sigma_k^+ + \Lambda_k^+), \quad (3)$$

where Σ_k^+ is the covariance that represents the online state estimation error, and Λ_k^+ is the covariance matrix of \hat{x}_k , which is a random variable when planning offline. It accounts for the uncertainty related to the measurements when

executed online. Analogous to the Kalman Filter, covariance matrices Σ_k^+ and Λ_k^+ can be computed recursively:

$$\Sigma_k^- = A\Sigma_{k-1}^+A^T + Q, \quad (4)$$

$$L_k = \Sigma_k^- C^T (C\Sigma_k^- C^T + R(\hat{x}_k))^{-1}, \quad (5)$$

$$\Sigma_k^+ = \Sigma_k^- - L_k C \Sigma_k^-, \quad (6)$$

$$\Lambda_k^+ = (A - BK)\Lambda_{k-1}^+(A - BK)^T + L_k C \Sigma_k^-. \quad (7)$$

Given belief b_k , its label $L(b_k)$ is the set of atomic propositions in Π that are true in b_k , i.e., $\pi_i^{\alpha_i} \in L(b_k) \iff P(x_k \in p_i) > 1 - \alpha$, where

$$P(x_k \in p_i) = \int_{p_i} \mathcal{N}(s | \tilde{x}_k, \Sigma_k^+ + \Lambda_k^+) ds. \quad (8)$$

Since our regions are convex polytopes that can be described as a conjunction of linear half-spaces, we can efficiently compute bounds for (8), using methods such as those described in [14], [15], [21], [32].

B. Task Planning with Pruned Automaton

At the task planning layer, an automaton is used to define a sequence of sub-tasks the robot has to complete.

1) *Automaton Construction and Pruning:* An LTLf formula ϕ can be translated into a minimized DFA that represents precisely the traces that satisfies ϕ [1]. A DFA is defined as a tuple $\mathcal{M} = (Q, q_0, S, \delta, F)$, where Q is a finite set of states, $q_0 \in Q$ is the initial states, $S = 2^\Pi$ is a set of input symbols, $\delta : Q \times S \rightarrow 2^Q$ is the transition function, $F \subseteq Q$ is the set of accepting states.

Instead of planning over a product automaton (Cartesian product of \mathcal{M} with a finite abstraction of the robotic system) like in [4], [5], [22], [24], [33], we propose to use the abstraction to prune impossible task sequences of \mathcal{M} . To do this, we construct an adjacency graph of the regions in \mathcal{P} in order to prune impossible transitions in \mathcal{M} .

The adjacency graph is constructed by computing adjacency and intersections between the regions in \mathcal{P} as well as the remainder region $X \setminus \cup_{p \in \mathcal{P}} p$. Each region is represented as a node in the adjacency graph, and edges of the graph represent regions which are geometrically adjacent to each other or intersecting with one another. This adjacency graph captures the possible transitions between regions in the continuous state space, and we use it to remove geometrically impossible letters in the alphabet, thereby pruning the edges in \mathcal{M} that are not realizable in the state space.

This approach has two advantages. First, it removes the need for fine abstractions of the problem space and system, which improves scalability by alleviating the state explosion problem. Our approach generalizes the method proposed by [6], by providing a way to prune the automaton using information about \mathcal{P} . Second, our method gives the belief planners in the subsequent layers the freedom to explore the belief space without restrictions, which is important for belief space planning, especially when the measurement noise is state dependent as in our problem, i.e., $\mathcal{R}(x_k)$.

Task planning is directly performed on \mathcal{M} augmented with edge weights that represent estimates on motion tree feasibility. The edge weighting scheme accounts for the fact that some transitions on the adjacency graph could be

geometrically possible (and thus exist in \mathcal{M}) but are not realizable by the system's belief dynamics. These weights are continuously updated during the planning process.

2) *Task Planning*: In each iteration of task planning, we compute an accepting run (task plan) \mathbf{T} on the pruned and weighted \mathcal{M} . A graph search algorithm, such as A^* , is used to find the shortest path to an accepting state in F from the initial state q_0 . Each task plan is a candidate sequence of automaton states that the algorithm uses to lead the search in the subsequent layers. By finding a belief trajectory that follows \mathbf{T} , the robot is guaranteed to satisfy ϕ .

To define the feasibility edge weights of \mathcal{M} , we first assign a weight to each state, similar to [4]. For state $q \in Q$,

$$w(q) = \frac{(\text{cov}(q)+1)}{\text{DistFromAcc}(q) \cdot (\text{numsel}(q)+1)^2}, \quad (9)$$

where $\text{cov}(q)$ is the number of motion tree vertices associated with q , $\text{numsel}(q)$ is the number of times q has been selected, and $\text{DistFromAcc}(q)$ is the shortest unweighted path from q to an accepting state in F . $\text{DistFromAcc}(q)$ can be computed using an unweighted graph search algorithm such as Dijkstra's Algorithm and can be computed once beforehand. For a minimized DFA, a path to an accepting state always exists from any state. Then, the edge weight between states $q, q' \in Q$ is $w(q, q') = (w(q) \cdot w(q'))^{-1}$. This weighting scheme promotes search in unexplored areas of the task space and suppresses search in areas where attempts at finding a solution have repeatedly failed.

C. Hybrid Tree Search: Gaussian Belief Trees with DFAs

For belief space planning with both the simplified model and full constrained model, we use Gaussian Belief Trees (GBT) in [19] as our base belief space planner in a hybrid discrete-continuous tree search planner. The hybrid planner is similar to the frameworks for deterministic LTL planners in [4], [5], [24], but it directly plans with automaton states instead of product automaton states. Also, it reasons about robot uncertainty while planning. For completeness of presentation, we provide a brief overview of this method.

Given a task plan \mathbf{T} , we compute the set of DFA states Q_T that exist in \mathbf{T} and contain at least one tree vertex at every iteration. The hybrid tree search expands in Q_T . In each iteration, a DFA state q is sampled from Q_T . A tree vertex v_s is sampled among vertices that have discrete components q , and one iteration of GBT is performed to obtain a new vertex v_n . A tree vertex is a tuple $v = (\tilde{x}, \Sigma, \Lambda, q)$. An iteration of GBT involves selecting a tree vertex v_{sel} , sampling a valid nominal control $\tilde{u} \in U$, and propagating the continuous belief components in v_{sel} to obtain the new belief $(\tilde{x}_n, \Sigma_n, \Lambda_n)$, according to (1) and Sec. II-B. The new discrete state component q_n of v_n is obtained by propagating the automaton with the label of the new belief.

Vertex v_n is valid and added to the tree if both its continuous and discrete state components are valid. The continuous component $v_n.b = (v_n.\tilde{x}, v_n.\Sigma, v_n.\Lambda)$ is valid if it obeys the constraints of (1). The discrete component $v_n.q$ is valid if it exists in the DFA. Finally, if $v_n.q$ is an

accepting state, the motion plan (\tilde{U}, \tilde{X}) and corresponding tree vertices v ending with v_n is returned as a solution.

D. SiMBA Layer for LTLf Guides

Once a task plan \mathbf{T} is generated, we can directly use the hybrid tree search with GBT on System (1) to attempt to satisfy the task sequence constraints, as described above. However, a naive method of tree search is computationally inefficient in the absence of a way to guide the search to promising areas of the large search space. To address this issue, we take inspirations from [21] and utilize trajectory biasing, and seek to rapidly find a path that satisfies the task plan. We propose to do this by using a simplified motion model of the system and include approximate belief dynamics (SiMBA). By simplifying the dynamics while still retaining some belief information, we are able to obtain a calculated trade-off between speed of finding a simplified solution path for guidance and informative guides in the belief space. Below, we formalize this idea for SiMBA.

First, we create a lower dimensional state space \tilde{X} which is a subspace of the state space X based on the projection $\tilde{x} = Proj(x)$ where $Proj: X \rightarrow \tilde{X}$ maps $x \in X$ to $\tilde{x} \in \tilde{X}$. This simplified state space can be arbitrarily chosen, with the only condition being that the new subspace \tilde{X} contains the subspace of all the polytopic regions \mathcal{P} , i.e., $p_i \subseteq \tilde{X} \forall p_i \in \mathcal{P}$. If the polytopic regions require all dimensions of the state space, SiMBA can still be used, albeit with $\tilde{x} = x$.

Second, we design a simplified motion model and measurement model for \tilde{x} . These models can also be arbitrarily chosen, but a general rule is to have simpler models for \tilde{x} than x , since the purpose of SiMBA is to compute approximate solutions rapidly in order to guide the belief search layer. Some generally applicable examples of simplified models are those with simple first order dynamics and kinematic models. One could also use simplifications such as: (i) geometric planning that ignores uncertainty, (ii) System (1) without the process and measurement noise terms, and (iii) System (1) without measurement noise.

For many robotic LTL tasks, the regions of interest are defined in the workspace. Therefore, we seek to design a motion model that generates kinematic trajectories. To do this, we take advantage of the observation that geometric paths can be interpreted as kinematic trajectories with a fixed speed in each direction. Thus, we use the maximum speed v_{max} of the robot. For the i th component of state \tilde{x} denoted by $\tilde{x}_k^{(i)}$, the motion model becomes:

$$\tilde{x}_{k+1}^{(i)} = \tilde{x}_k^{(i)} + v_i, \quad (10)$$

such that $\sum v_i^2 = v_{max}^2$. As a simple approximation, we maintain the covariance propagation of the original system using (4)-(7). We refer to this SiMBA as Simple Belief Approximation-SiMBA (SBA-SiMBA).

Definition 2 (Admissible SiMBA). *Consider System (1) with X and its subspace \tilde{X} for a SiMBA. The SiMBA is admissible if, for every instance of Problem 1 that admits a solution for System (1), a solution also exists for the SiMBA in \tilde{X} .*

Since the purpose of SiMBA is to find a coarse solution quickly to guide the belief search layer, it is desirable to use admissible SiMBAs. However, the algorithm can still be probabilistically complete with an inadmissible SiMBA, as we discuss in Sec. IV.

In many robotics scenarios, low robot uncertainty is more desirable than high uncertainty. The intuition is that large uncertainty leads to higher risks (e.g., collision probability). However, mathematically, when bad regions (e.g., obstacles) are small, large uncertainty can reduce risk. To rule out such unintuitive cases, we make the following assumption, which we also use to build our SiMBA methodology.

Assumption 1. If $b_a = \mathcal{N}(\tilde{x}, \Sigma_a)$, $b_b = \mathcal{N}(\tilde{x}, \Sigma_b)$, and $\Sigma_a < \Sigma_b$, then

- for all $\alpha < 0.5$, $\pi^\alpha \in L(b_b) \implies \pi^\alpha \in L(b_a)$, and
- for all $\alpha > 0.5$, $\pi^\alpha \notin L(b_b) \implies \pi^\alpha \notin L(b_a)$

Assumption 1 states that increasing the covariance does not increase the probability of being in a region at a given state mean for low α , which corresponds to ‘reach’ regions. Additionally, increasing the covariance does not increase the probability of not being in a region at a given state mean for high α , which corresponds to ‘avoid’ regions.

Lemma 1. SBA-SiMBA is admissible under Assumption 1.

Proof Sketch. Starting from the same \tilde{x}_0 and the same covariance, SBA-SiMBA is able to under-approximate the uncertainty of the belief of the original system at any \tilde{x} , since there are less constraints on its dynamics and it uses the maximum speed of the original system. From Assumption 1, a lower covariance at each state leads to better probability of satisfaction. There exists at least as many accepting belief trajectories for SiMBA as for the full system. \square

Finally, the goal is to find a feasible solution for the SiMBA augmented version of Problem 1 in each iteration of the planning framework. Using the hybrid search tree in Sec. III-C, we obtain a kinematic trajectory for the simplified space \tilde{X} . A SiMBA path is then the hybrid path $\xi_{\tilde{X}} = (\tilde{x}, \mathbf{q})$ which is a sequence of pairs of \tilde{x} and \mathcal{M} state q .

We note that designing simplified models can generally be difficult, but our results show that using SBA-SiMBA significantly reduces planning time for various LTLf tasks.

E. Belief Search Layer

The belief search layer plans for the original constrained System (1) using GBT to find a solution for Problem 1. To guide the search, we lift SiMBA path $\xi_{\tilde{X}}$ from \tilde{X} to X and use biased sampling around the lifted trajectory set.

We propose to guide the belief tree search by using the *sub-task relevant* segments of $\xi_{\tilde{X}}$. Let v be an existing tree node in the belief search layer with discrete component $v.q_i$, and let q_{i+1} be the successor of q_i in task plan \mathbf{T} . Then, we use the segment of $\xi_{\tilde{X}}$ with discrete components q_i and those that lead to a transition from q_i to q_{i+1} , i.e., the segment is of the form $\xi_{\tilde{X}}^{q_i} = (\tilde{x}_k, q_i) \dots (\tilde{x}_{k+m}, q_i) (\tilde{x}_{k+m+1}, q_{i+1})$, where k is the first time step $\xi_{\tilde{X}}$ encounters q_i and $m \geq 0$.

The sampling bias technique is as follows. Given a starting radius d which is incrementally increased, samples are chosen within d radius of $\xi_{\tilde{X}}^{q_i}$ with probability pr , and uniformly in the belief space with probability $1 - pr$. Biasing sampling in this way allows the belief search tree to bias growth towards promising parts of the belief space found by the SiMBA layer while still allowing exploration.

At each iteration, both SiMBA guide and belief search layers are given a user-defined time bound to extend the belief tree. The intuition is to continually feedback feasibility information from the tree search layers to the task planning layer. Weights on \mathcal{M} are updated based on (9) and the algorithm continues to the next iteration.

IV. THEORETICAL GUARANTEES

Here, we analyze the theoretical guarantees of our proposed framework.

Theorem 1 (Correctness). Let $V^* = (v_0, \dots, v_f)$ be a returned solution trajectory from the belief search layer. Then, the observation trace of its continuous component is guaranteed to satisfy ϕ , i.e., $\sigma \models \phi$.

Proof Sketch. V^* is only returned as a solution if $v_f.q \in F$. Given polytopic regions \mathcal{P} , the computation of chance constraints for labels in (8) is conservative. Since the belief covariance ($\Sigma + \Lambda$) is computed exactly, the label is conservatively computed, i.e., $L(b_k)$ is correct. Thus, the belief components of V^* (belief trajectory \mathbf{b}) correctly returns an accepting run on \mathcal{M} , so \mathbf{b} satisfies ϕ , i.e., $\mathbf{b} \models \phi$. \square

Theorem 2 (Probabilistic Completeness). Given an admissible SiMBA, our framework is probabilistically complete.

Proof Sketch. This follows from Thm 1, the definition of admissible SiMBA, and the results of [19]. \mathcal{M} ’s edge weights are continually updated according to feasibility estimates, so every accepting run in \mathcal{M} will be sampled infinitely often in the limit. In the limit, the search space will be covered by infinitely dense trees for both SiMBA guide and belief search layers, from the probabilistic completeness of GBT. From Thm 1, any returned solution is guaranteed to be correct. \square

Note that we can still achieve probabilistic completeness with an inadmissible SiMBA, by setting a separate time bound to the SiMBA layer and conducting belief search without a SiMBA guide if no solution is found by the SiMBA layer. However, admissible SiMBA guides can greatly reduce computation times, as seen in our evaluation.

V. EVALUATIONS

We demonstrate the effectiveness of our proposed framework, through benchmarks and illustrative case studies.

Simulation Benchmarks: We compare our proposed framework with (i) a state-abstraction based approach (Abs-based), similar to extending [5] to the belief space by using a belief space planner, and (ii) using our framework but without a SiMBA layer as an ablation study, and using the full framework with different SiMBAs, (iii) a geometric SiMBA that plans directly in the state space (Geo-SiMBA),

TABLE I: Benchmarking results for underwater inspection scenario. We report the mean time to solution and standard error of the mean. Runs are given a maximum time of 120s, with successful runs finding a solution within 120s. The best scores are shown in bold font.

Algorithm	ϕ_1		ϕ_2		ϕ_3		ϕ_4		ϕ_5	
	Succ. (%)	Time (s)	Succ. (%)	Time (s)	Succ. (%)	Time (s)	Succ. (%)	Time (s)	Succ. (%)	Time (s)
Abs-based	0	NA	0	NA	0	NA	0	NA	0	NA
Simba-free	100	7.7 \pm 0.8	66	71.6 \pm 4.3	63	78.7 \pm 4.0	14	114.1 \pm 1.7	4	117.7 \pm 1.2
Geo-SiMBA	100	6.9 \pm 0.6	89	42.9 \pm 3.6	81	59.3 \pm 4.2	39	103.5 \pm 2.6	32	107.3 \pm 2.4
SBA-SiMBA	100	5.8 \pm 0.5	95	29.3 \pm 2.9	92	34.9 \pm 3.2	73	85.4 \pm 3.0	54	94.5 \pm 3.1

and (iv) SBA-SiMBA which includes belief approximation as discussed in Sec. III-D.

The scenario is a simplified underwater cave inspection with obstacles as depicted in Figure 2. Here, the robot can receive measurements with small noise about its state near the surface (in white), while it has to rely on noisier IMU otherwise. The robot has dynamics given by $\dot{x} = v \cos(\theta)$, $\dot{y} = v \sin(\theta)$, $\dot{\theta} = \omega$, $\dot{v} = a$. We use state feedback linearization to obtain a linear closed-loop model as in [21], [34]. We consider the following increasingly complex LTLf formulas, with $a = \pi_a^{0.95}$ for region A (in green), $b = \pi_b^{0.95}$ for region B (in yellow), $c = \pi_c^{0.95}$ for region C (in white), and $o = \tilde{\pi}_o^{0.95}$ for region O (brown and black obstacles):

- $\phi_1 = \square \neg o \wedge \diamond a$
- $\phi_2 = \square \neg o \wedge \diamond (a \wedge \diamond c)$
- $\phi_3 = \square \neg o \wedge \diamond (a \wedge \diamond c) \wedge \diamond (b \wedge \diamond c)$
- $\phi_4 = \square \neg o \wedge \diamond (a \wedge \diamond (c \wedge \diamond (a \wedge \diamond c)))$
- $\phi_5 = \square \neg o \wedge \diamond (a \wedge \diamond (c \wedge \phi_3)) \wedge \diamond (b \wedge \diamond (c \wedge \phi_3))$

For each specification, $\square \neg o$ refers to obstacle avoidance. ϕ_1 requires to eventually reach region a . ϕ_2 is a sequential goal problem, with the goal of reaching region a followed by region c . ϕ_3 states that the robot should visit regions a and b in any order, before surfacing to region c . ϕ_4 states that the robot should visit regions a followed by c twice. ϕ_5 states that the robot should satisfy ϕ_3 twice sequentially.

For belief tree planning, we used [19] with RRT. We provided a time limit of 120s to find a solution for 100 trials. The results are summarized in Table I. From the benchmarking results, it is evident that the state-abstraction approach does not work for belief space problems. This is due to the need to capture measurements affecting the belief dynamics in the abstraction. Note that it may be possible to combine abstraction-based approaches with a biased sampling paradigm, but it is not clear how to do so effectively. We also attempted to use belief discretization instead of state discretization, but it also performed poorly since many transitions in the abstraction are impossible.

Our results show that as complexity of specifications increases, so does the benefit of our planning scheme and the importance of SiMBA guides. Even for reach-avoid task ϕ_1 , our planning scheme with both types of SiMBAs show better performance compared to the other methods, demonstrating the general efficacy of this approach for belief space planning with LTLf specifications. Furthermore, the addition of belief approximation in SiMBA allows for more informative guides that reason about the belief space, leading to better solution times for SBA-SiMBA as compared to Geo-SiMBA. Typical solution trajectories returned by our proposed algorithm are shown in Figure 2, where the robot intelligently navigates

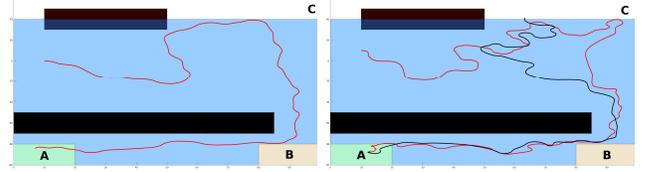


Fig. 2: Underwater inspection. ϕ_1 (left): the robot moves close to the surface to localize before completing the task. ϕ_2 (right): the robot completes sub-task $\diamond a$ (red) and nested sub-task $\diamond c$ (black).

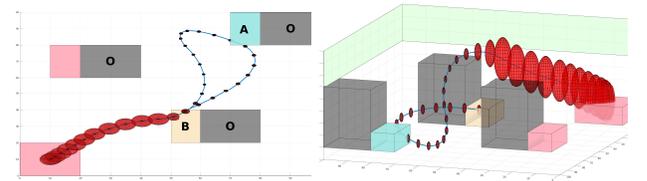


Fig. 3: Drone delivery, top down (left) and side (right) views.

close to the surface to localize (reduce uncertainty) before submerging and completing its tasks in the cave, through which measurements have higher noise values.

Urban Drone Delivery: Next, we demonstrate our algorithm in a simulated drone delivery problem in an urban environment, as depicted in Figure 3. Details on the dynamics can be found in Appendix A.2 of [21]. We consider a delivery problem, where UAV has to pick up a package at the foot of a building A (region A in blue), and deliver it to foot of building B (region B in yellow) while avoiding obstacles (in gray). Additionally, it can only receive GPS measurements if it flies above the gray buildings (in light green), but it cannot fly that high after picking up a package. In LTLf, this task is $\phi_6 = \square \neg o \wedge \diamond (a \wedge \diamond b) \wedge \square (a \implies \square \neg c)$, with $a = \pi_a^{0.99}$, $b = \pi_b^{0.99}$, $c = \tilde{\pi}_c^{0.99}$, and $o = \tilde{\pi}_o^{0.99}$. We conducted 100 trials for this problem with time limit of 120s, and our algorithm with SBA-SiMBA found a solution with 90% success rate, with mean time to solution of 30 ± 4.1 . A typical solution is shown Figure 3. The UAV first flies above the buildings to reduce its uncertainty, before flying to region A and then B, to ensure that its uncertainty is low enough to reach both regions with high probability.

VI. CONCLUSION AND FUTURE WORK

We presented a modular framework for motion planning with complex tasks for systems under both motion and measurement uncertainty. We showed that simplified belief models provide important information for guiding motion tree search for planning in belief space. Our proposed framework is correct by construction and probabilistically complete. Empirical evaluations demonstrate the efficiency and efficacy of the planner. Future work aims at analyzing how to design accurate simplified models for better guides.

REFERENCES

- [1] G. De Giacomo and M. Y. Vardi, "Linear temporal logic and linear dynamic logic on finite traces," in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, ser. IJCAI '13. AAAI Press, 2013, p. 854–860.
- [2] H. Kress-Gazit, M. Lahijanian, and V. Raman, "Synthesis for robots: Guarantees and feedback for robot behavior," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 211–236, 2018. [Online]. Available: <https://doi.org/10.1146/annurev-control-060117-104838>
- [3] G. Fainekos, H. Kress-Gazit, and G. Pappas, "Temporal logic motion planning for mobile robots," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 2020–2025.
- [4] A. Bhatia, L. E. Kavraki, and M. Y. Vardi, "Sampling-based motion planning with temporal goals," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 2689–2696.
- [5] M. R. Maly, M. Lahijanian, L. E. Kavraki, H. Kress-Gazit, and M. Y. Vardi, "Iterative temporal motion planning for hybrid systems in partially unknown environments," in *International Conference on Hybrid Systems: Computation and Control*, 2013, p. 353–362.
- [6] X. Luo, Y. Kantaros, and M. M. Zavlanos, "An abstraction-free method for multirobot temporal logic optimal control synthesis," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1487–1507, 2021.
- [7] E. Plaku and S. Karaman, "Motion planning with temporal-logic specifications: Progress and challenges," *AI Commun.*, vol. 29, no. 1, pp. 151–162, 2016. [Online]. Available: <https://doi.org/10.3233/AIC-150682>
- [8] A. M. Wells, M. Lahijanian, L. E. Kavraki, and M. Y. Vardi, "LTLf Synthesis on Probabilistic Systems," *arXiv e-prints*, p. arXiv:2009.10883, Sep. 2020.
- [9] X. C. Ding, J. Wang, M. Lahijanian, I. C. Paschalidis, and C. A. Belta, "Temporal logic motion control using actor-critic methods," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 4687–4692.
- [10] R. Luna, M. Lahijanian, M. Moll, and L. E. Kavraki, "Asymptotically optimal stochastic motion planning with temporal goals," in *Algorithmic Foundations of Robotics XI - Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics, WAFR*, ser. Springer Tracts in Advanced Robotics, vol. 107. Springer, Aug. 2015, pp. 335–352.
- [11] M. Bouton, J. Tumova, and M. J. Kochenderfer, "Point-based methods for model checking in partially observable markov decision processes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, Apr. 2020, pp. 10061–10068. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/6563>
- [12] S. Haesaert, P. Nilsson, C. Vasile, R. Thakker, A. Agha-mohammadi, A. Ames, and R. Murray, "Temporal logic control of pomdps via label-based stochastic simulation relations," in *6th IFAC Conference on Analysis and Design of Hybrid Systems 2018*, 2018, pp. 271–276.
- [13] Z. Kingston, M. Moll, and L. E. Kavraki, "Sampling-based methods for motion planning with constraints," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 159–185, 2018.
- [14] B. Luders, M. Kothari, and J. How, "Chance constrained rt for probabilistic robustness to environmental uncertainty," in *AIAA Guidance, Navigation, and Control Conference*, 2010. [Online]. Available: <https://arc.aiaa.org/doi/abs/10.2514/6.2010-8160>
- [15] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, 2011.
- [16] W. Liu and M. H. Ang, "Incremental sampling-based algorithm for risk-aware planning under motion uncertainty," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 2051–2058.
- [17] A.-a. Agha-mohammadi, S. Chakravorty, and N. M. Amato, "Firm: Feedback controller-based information-state roadmap - a framework for motion planning under uncertainty," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 4284–4291.
- [18] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *International Conference on Robotics and Automation*. IEEE, 2011.
- [19] Q. H. Ho, Z. N. Sunberg, and M. Lahijanian, "Gaussian belief trees for chance constrained asymptotically optimal motion planning," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 11 029–11 035.
- [20] D. Zheng, J. Ridderhof, P. Tsiotras, and A.-a. Agha-mohammadi, "Belief space planning: a covariance steering approach," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 11 051–11 057.
- [21] E. Pairet, J. D. Hernandez, M. Carreras, Y. Petillot, and M. Lahijanian, "Online mapping and motion planning under uncertainty for safe navigation in unknown environments," *IEEE Transactions on Automation Science and Engineering*, pp. 1–23, 2021.
- [22] Y. Oh, K. Cho, Y. Choi, and S. Oh, "Chance-constrained multilayered sampling-based path planning for temporal logic-based missions," *IEEE Transactions on Automatic Control*, vol. 66, no. 12, pp. 5816–5829, 2021.
- [23] E. Plaku, L. E. Kavraki, and M. Y. Vardi, "Motion planning with dynamics by a synergistic combination of layers of planning," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 469–482, 2010.
- [24] Q. H. Ho, R. B. Ilyes, Z. N. Sunberg, and M. Lahijanian, "Automaton-guided control synthesis for signal temporal logic specifications," 2022. [Online]. Available: <https://arxiv.org/abs/2207.03662>
- [25] P. Tajvar, F. S. Barbosa, and J. Tumova, "Safe motion planning for an uncertain non-holonomic system with temporal logic specification," in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, 2020, pp. 349–354.
- [26] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 11 2005. [Online]. Available: <https://doi.org/10.7551/mitpress/3206.001.0001>
- [27] D. Sadigh and A. Kapoor, "Safe control under uncertainty with probabilistic signal temporal logic," in *Proceedings of Robotics: Science and Systems*, Ann Arbor, Michigan, June 2016.
- [28] R. R. da Silva, V. Kurtz, and H. Lin, "Active perception and control from prstl specifications," 2021. [Online]. Available: <https://arxiv.org/abs/2111.02226>
- [29] S. Jha, V. Raman, D. Sadigh, and S. A. Seshia, "Safe autonomy under perception uncertainty using chance-constrained temporal logic," *J Autom Reasoning*, vol. 60, no. 1, p. 43–62, jan 2018. [Online]. Available: <https://doi.org/10.1007/s10817-017-9413-9>
- [30] Y. Kantaros, S. Kalluraya, Q. Jin, and G. J. Pappas, "Perception-based temporal logic planning in uncertain semantic maps," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2536–2556, 2022.
- [31] N. Cauchi, L. Laurenti, M. Lahijanian, A. Abate, M. Kwiatkowska, and L. Cardelli, "Efficiency through uncertainty: Scalable formal synthesis for stochastic hybrid systems," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, ser. HSCC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 240–251. [Online]. Available: <https://doi.org/10.1145/3302504.3311805>
- [32] W. Liu and M. H. Ang, "Incremental sampling-based algorithm for risk-aware planning under motion uncertainty," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 2051–2058.
- [33] Y. Kantaros and M. M. Zavlanos, "Stylus*: A temporal logic optimal control synthesis algorithm for large-scale multi-robot systems," *The International Journal of Robotics Research*, vol. 39, no. 7, pp. 812–836, 2020. [Online]. Available: <https://doi.org/10.1177/0278364920913922>
- [34] A. De Luca, G. Oriolo, and M. Vendittelli, "Stabilization of the unicycle via dynamic feedback linearization," *IFAC Proceedings Volumes*, vol. 33, no. 27, pp. 687–692, 2000.