Toward Specification-Guided Active Mars Exploration for Cooperative Robot Teams

Petter Nilsson*, Sofie Haesaert*, Rohan Thakker†, Kyohei Otsu†, Cristian-Ioan Vasile‡,
Ali-akbar Agha-mohammadi†, Richard M. Murray*, and Aaron D. Ames*
*California Institute of Technology, Pasadena, CA 91125
†NASA Jet Propulsion Laboratory, Pasadena, CA 91109
‡Massachusetts Institute of Technology, Cambridge, MA 02139

***Abstract*—As a step towards achieving autonomy in space exploration missions, we consider a cooperative robotics system consisting of a copter and a rover. The goal of the copter is to explore an unknown environment so as to maximize knowledge about a science mission expressed in linear temporal logic that is to be executed by the rover. We model environmental uncertainty as a belief space Markov decision process and formulate the problem as a two-step stochastic dynamic program that we solve in a way that leverages the decomposed nature of the overall system. We demonstrate in simulations that the robot team makes intelligent decisions in the face of uncertainty.**

I. Introduction

Environment exploration and task planning are crucial components of any autonomous system. In most approaches, these two aspects are decoupled in the sense that exploration is performed to maximize knowledge overall, rather than to maximize knowledge *pertaining to the task*. In this work, we aim to improve efficiency in situations where exploration is expensive by limiting exploration to areas crucial for accomplishing a given task.

Due to the growing complexity and uncertainty in future space missions, autonomy is a crucial ability required for mission success. We focus on multi-asset missions (teams of robots), in particular the 2020 Mars mission that will consist of a ground rover and a copter for exploration (Fig. 1). To increase productivity and science return, the robotic team needs to autonomously perform multi-sol (sol: Martian day) navigation without human intervention. Severe communication delays and resource constraints (such as battery time and limited hours of sunlight) pose further challenges in achieving such autonomy. In these missions, partial knowledge about the environment is typically available from satellite imagery, but it needs to be complemented with observations from on-board sensors. Our goal is to improve both autonomy and efficiency by developing principled methods that determine the most important areas to explore in a specification-guided manner. We focus specifically on the problem of determining how the copter should behave to assist the rover in the autonomous execution of mission tasks.

Our approach lies at the intersection of formal synthesis methods and stochastic optimal control; we phrase the design problem as a two-stage optimal control problem in the combined space of rover-copter poses and environment beliefs. To partially circumvent the curse of dimensionality we leverage the decomposed nature of the problem and perform value iteration via sequential back-stepping, which avoids explicit construction of the large aggregate system.

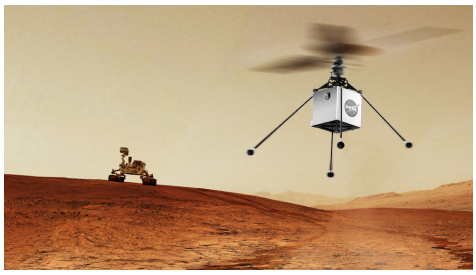

Fig. 1. Cooperative robotics team for Mars exploration.

Related Work. Environment mapping for robot navigational purposes is an important prerequisite for robotic autonomy for which many types of methods have been proposed [27, 29, 30]. While mapping algorithms can be developed separately from planning algorithms, in this work, we are interested in *joint* planning and mapping methods. Active perception techniques fall into this category [4, 7, 26]. Active mapping or SLAM algorithms, a subcategory of active perception, aims to enhance the quality of the environment map (e.g., for the navigation task [3, 10, 12, 24]) by choosing information-rich trajectories. However, these methods are typically limited to subsystem autonomy (e.g., navigation) and do not have a principled way of incorporating system-level constraints and high-level mission specifications and requirements.

Motion planning and control of robots tasked with missions expressed as linear temporal logic has been addressed in both deterministic [16, 18, 22, 31] and stochastic [20, 23, 28] settings. Active estimation has been investigated in [15, 21], and quadcopter-aided exploration and pose estimation was proposed in [9]. However, specification-guided cooperative control with agents that have heterogeneous roles has to the best of our knowledge not been explored in previous work.

Contributions. In this work, we aim to develop a novel methodology that allows for principled incorporation of mission specifications where the exploration not only helps navigation, but also assists with satisfaction of mission goals.

The contributions are threefold. Firstly, in Section II we construct a novel problem formulation for future multi-asset Mars missions by modeling all system components as Markov

decision processes, and by expressing science objectives using temporal logics. Secondly, in Section III we show that the exploration task can be reformulated as a stochastic optimal control problem that is specific to the rover task. In Section IV, we introduce—as the third contribution—computational methods that leverage the inherent decomposed structure of the problem to mitigate the curse of dimensionality. Additionally in Section V, we illustrate these new concepts on a case study that exhibits typical aspects of a Mars exploration problem, before concluding the paper in Section VI.

II. Problem Setup

A. Markov Decision Process Models

We model the rover and copter as well as environment uncertainty as Markov decision processes [6].

Definition 1. *A discrete-time* ***Markov decision process*** *(MDP) is a tuple* $\mathbb{M} = (\mathbb{X}, x_0, \mathbb{U}, T)$ *where* $\mathbb{X}$ *is a state space with states* $x \in \mathbb{X}$*;* $x_0 \in \mathbb{X}$ *is the initial state;* $\mathbb{U}$ *is an input space with inputs* $u \in \mathbb{U}$*; and* T *is a conditional stochastic kernel that assigns to each state* $x \in \mathbb{X}$ *and control input* $u \in \mathbb{U}$ *a probability distribution* $T(\cdot \mid x, u)$ *over* $\mathbb{X}$*.*

For a given sequence of control inputs $u_t \in \mathbb{U}$, we say that an *execution* of $\mathbb{M}$ is a sequence of states $\mathbf{x} = x_0 x_1 x_2 x_3 \ldots$ such that $x_{t+1} \sim T(\cdot \mid x_t, u_t)$ for $t \geq 0$. A *policy* $\mu = \mu_0 \mu_1 \ldots$ is a sequence of mappings $\mu_t : \prod_t \mathbb{X} \to \mathbb{U}$ from the history space to the action space. An execution is controlled by μ if $x_{t+1} \sim T(\cdot \mid x_t, \mu_t(x_0, \ldots, x_t))$. A policy is *Markov* if for all t, $\mu_t(x_0, \ldots, x_t) = \mu_t(x_t)$. All systems in this paper are modeled as MDPs, or as combinations (products) of MDPs.

Rover and Copter as MDPs. In case of the rover, the state space $\mathbb{X}$ represents the two-dimensional[1] space in which the rover navigates. Its state transitions can be modeled as being either stochastic or deterministic. Similarly, the dynamics of the copter can be represented in three-dimensional space, where the additional dimension represents altitude. Effects of wind and other uncertainty influencing the state transitions can be captured by the stochastic transitions in the copter MDP model. For brevity, we refer to the rover MDP as $\mathbb{M}_{rov}$ with state variable x^r and to the MDP model of the copter as $\mathbb{M}_{cop}$ with state x^c.

Environment Uncertainty as a Belief MDP. A common way to represent a map for planning purposes is to attach labels such as "sand" or "rock" to different *regions*, but in a Mars setting the true labels are often only partially known at the time of system deployment.

We capture this environmental uncertainty with MDPs $\mathbb{M}_{e_i}$ that model belief dynamics over labeled regions i of the two-dimensional rover state space. Thus, the state x^{e_i} of $\mathbb{M}_{e_i}$ takes values between 0 and 1 and corresponds to the estimated probability that region i has a certain label. The aggregate environment model $\mathbb{M}_{env}$ with state x^e is the combination of individual MDPs $\mathbb{M}_{e_i}$. Transitions in these MDPs occur when measurements are taken from on-board sensors, i.e., the stochastic kernels are parameterized by the physical distance of the robots to the regions in a measurement model. For instance, as illustrated in Fig. 2 it is reasonable to assume that the rover can take measurements of a given region provided that it is sufficiently close, and that the copter can take measurements of regions it flies above. Copter measurements also depend on the altitude: a high-flying copter can see a larger area but the image resolution is better if the copter is close to the ground. Hence the measurement models dictate the structured composition of the copter and rover models with the environment model. The resulting composed MDP represents the full model of the state of both robots and the current knowledge of the environment.

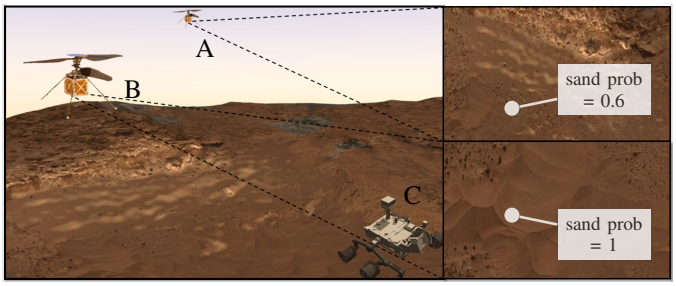

Fig. 2. Taking measurements at higher altitudes (A) allows the helicopter to cover larger area, however, due to low resolution of the image the inferred labels have a larger uncertainty. Whereas, taking measurements at lower altitudes (B) results in lower uncertainty of the measurements but a smaller coverage area. Finally, the rover (C) can get measurements with very low uncertainty as it gets closer to the terrain. Further, it can also get the true label using its proprioceptive and exteroceptive sensors. For example, sand slippage can be detected by measuring discrepancies between location estimates from wheel odometery and from visual odometry.

B. Formal Specifications

We use temporal logic formulas to formally describe tasks that the rover should perform. The basic building blocks of a temporal logic formula are *atomic propositions* that take values `true` or `false`. An atomic proposition p is associated with a subset A of a state space $\mathbb{X}$ in the sense that $p = \texttt{true}$ if and only if $x \in A$.

By combining atomic propositions into a formula, desired system behavior is expressed as set membership constraints (atomic propositions) together with temporal relations (operators). Consider a set $AP = \{p_1, \ldots, p_L\}$ of atomic propositions; it defines an *alphabet* 2^{AP} where each *letter* $\pi \in 2^{AP}$ of the alphabet is a set of atomic propositions. An infinite string of letters is a *word* $\boldsymbol{\pi} = \pi_0 \pi_1 \pi_2 \ldots$, which should be thought of as the observed output of a system. More specifically, for an execution $\mathbf{x} = x_0 x_1 x_2 \ldots$ of an MDP, a labeling function $L : \mathbb{X} \to 2^{AP}$ that maps states to outputs yields the associated word as $\boldsymbol{\pi} = L(x_0)L(x_1)L(x_2)\ldots$. Desired behavioral properties can now be expressed via temporal logic formulas over the generated words. In the sequel, we focus on a fragment of linear temporal logic.

[1]The rover moves slowly and can turn on the spot, so yaw dynamics can be ignored.

Definition 2. *Formulas in the* ***syntactically co-safe LTL*** *(scLTL) fragment are constructed according to the grammar*

$$\varphi ::= \mathsf{true} \,|\, p \,|\, \neg p \,|\, \varphi_1 \vee \varphi_2 \,|\, \varphi_1 \wedge \varphi_2 \,|\, \varphi_1 \mathcal{U} \varphi_2 \,|\, \bigcirc \varphi, \quad (1)$$

where $p \in AP$ is an atomic proposition.

The syntax (1) defines the symbols and their correct ordering in a formula. In contrast, the semantics defined next give the interpretation of a formula.

Definition 3. *We write $(\boldsymbol{\pi}, t) \models \varphi$ to indicate that a word $\boldsymbol{\pi}$* ***satisfies*** *φ at time t. Satisfaction is defined recursively as follows: $(\boldsymbol{\pi}, t) \models \mathsf{true}$; $(\boldsymbol{\pi}, t) \models p$ iff $p \in \pi_t$; $(\boldsymbol{\pi}, t) \models \varphi_1 \wedge \varphi_2$ iff $((\boldsymbol{\pi}, t) \models \varphi_1) \wedge ((\boldsymbol{\pi}, t) \models \varphi_2)$; $(\boldsymbol{\pi}, t) \models \varphi_1 \vee \varphi_2$ iff $((\boldsymbol{\pi}, t) \models \varphi_1) \vee ((\boldsymbol{\pi}, t) \models \varphi_2)$; $(\boldsymbol{\pi}, t) \models \varphi_1 \mathcal{U} \varphi_2$ iff $\exists s \geq t$ s.t. $((\boldsymbol{\pi}, s) \models \varphi_2)$ and $(\boldsymbol{\pi}, l) \models \varphi_1, \forall l \in \{t, \dots s-1\}$; $(\boldsymbol{\pi}, t) \models \bigcirc \varphi$ iff $(\boldsymbol{\pi}, t+1) \models \varphi$.*

We say that a state trajectory $\mathbf{x} = x_0 x_1 x_2 \dots$ satisfies a specification φ, written $\mathbf{x} \models \varphi$, if the generated word $\boldsymbol{\pi} = L(x_0)L(x_1)L(x_2)\dots$ satisfies φ at time 0, i.e. $(\boldsymbol{\pi}, 0) \models \varphi$. We use the shorthand notation $\Diamond a$—*eventually* a—to express the property that a set labeled with a should eventually be reached, i.e. $\Diamond a = \mathsf{true}\, \mathcal{U}\, a$.

Since transitions are stochastic we can in general not say that an MDP $\mathbb{M}$ satisfies a property φ. We can however for a given policy μ quantify the *probability* that the system satisfies the property, i.e. compute or approximate

$$\mathbb{P}_{\mu}^{\mathbb{M}}[\mathbf{x} \vDash \varphi]. \quad (2)$$

For our purposes, the interesting scenarios are when there is a lot of uncertainty about whether a mission can be completed or not. If the mission is trivial, or it is already known to be difficult, exploration will likely only further establish these beliefs. In cases where mission completion is uncertain, we want to leverage the copter to reduce *mission risk*: the probability that a mission we choose to undertake ultimately fails. Having introduced MDP models for rover, copter, and environment belief, as well as scLTL specifications, the problem we address in this paper can be stated as follows.

Problem 1. *Consider a robot-copter team in an uncertain environment, all modeled as MDPs. Design a strategy for the robot-copter team that reduces the probability of failure of a scientific mission given as an scLTL formula.*

We have chosen to work with the scLTL fragment that is restricted to properties that can be satisfied in finite time. For the purpose of Mars exploration where tasks are often defined on a daily basis (and thus are of finite duration) this fragment is capable of representing a large class of relevant specifications.

C. Problem Decomposition

Due to the nature of the mission—the copter is quick and has short battery life (minutes), the rover moves slowly (over hours)—it is reasonable to divide the problem into two subproblems. We propose to solve the problem sequentially based on its natural division into two phases: exploration and mission execution. As shown in Fig. 3, the copter is active in the exploration phase, whereas the rover conducts the critical scientific mission in the second phase. We assume that the exploration phase lasts for a time T_c, and that the mission phase lasts at most for a time T_r. These times can be determined based on the available battery capacity and remaining hours of daylight. In both phases the environment belief is updated according to measurements taken by the active agent. While the execution order of the phases is exploration followed by mission execution, the reverse holds for design: first the mission design problem is solved and its solution informs the exploration design.

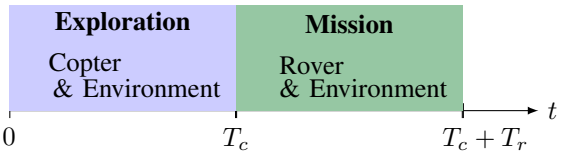

Fig. 3. Illustration of execution order. In the exploration phase $[0, T_c]$ the copter explores the environment, whereas the rover executes the scientific mission on $[T_c, T_c + T_r]$.

Mission Problem. The objective in the mission phase is to maximize the probability that the specification φ is satisfied, thus part of the Mission Problem is to synthesize such a policy. However, to guide the design of an exploration policy it is also necessary to quantify the probability of success for different environment belief states at time T_c. To this end, the objective of the Mission Problem is as follows: for a given initial rover state $x^r_{T_c}$ and for all environment belief states $x^e_{T_c}$ compute the probability that the mission specification is satisfied, along with a maximizing policy.

Exploration Problem. Our objective is to extract control policies that maximize the knowledge about the satisfiability of a given task: either we want the exploration to show that the task can be completed with a high probability, or that the probability of task completion is very low. Both are desirable outcomes since a negative result allows resources to be redirected toward more realistic objectives. Based on the partition into exploration and mission phases we refine our problem as follows.

Problem 2. *Consider the model components listed above. Design a policy for the copter exploration phase that explores the environment in a way that maximizes the expected knowledge about satisfaction of the scientific mission at time T_c.*

Computational Challenge. Due to the multiple interacting model components, a solution to Problem 2 is potentially computationally challenging. The decomposition into an exploration phase and a mission phase already improves scalability significantly by limiting the number of systems that are active at any given time (c.f. Fig. 3). We additionally show in Section IV how the problem can be solved sequentially over system components, which avoids the expensive step of explicit computation of transition probabilities in the aggregate systems. Combined, these two properties allow us to synthesize policies for realistic problems of modest size.

III. A Stochastic Optimal Control Approach

We now detail how both the Mission Problem and the Exploration Problem can be phrased as optimal reachability problems.

A. Optimal Control Solution of Mission Problem

We first show that specification satisfaction can be formulated as a stochastic reachability problem over an extended MDP, and then detail how the solution can be obtained via stochastic finite-horizon dynamic programming. We start with an example illustrating how extended systems can be used to reason about specification satisfaction.

Example 1. *Consider the example in Fig. 4 consisting of an example MDP and a mission specification* $\varphi := \Diamond a \wedge \Diamond b$ *dictating that two tasks* a *and* b *should both eventually be satisfied, for* $L(x^a) = \{a\}$, $L(x^b) = \{b\}$, *and* $L(x^c) = \emptyset$. *The automaton system to the right in Fig. 4 has the property that a trace* $\pi = L(x_0)L(x_1)L(x_2)\ldots$ *satisfies the specification if and only if the sequence of inputs* π *causes the system to reach the target state* q_f. *It follows that a trajectory of the MDP to the left satisfies* φ *if and only if the corresponding joint execution of the MDP and the automaton reaches the final state* q_f.

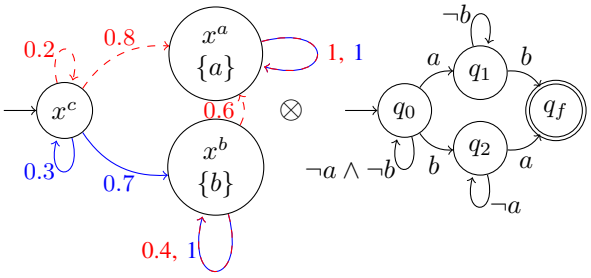

Fig. 4. On the left an example MDP, and on the right a finite-state automaton corresponding to the mission specification $\Diamond a \wedge \Diamond b$.

The procedure in Example 1 is general: for any specification φ in the scLTL fragment it is possible to construct a finite-state automaton $\mathcal{A}_\varphi$ with accepting set Q_f [5]. The probability (2) that a specification is satisfied for an MDP $\mathbb{M}$ with state space $\mathbb{X}$ is thus equivalent to the probability that a set $\mathbb{X} \times Q_f$ in the state space of the extended product system $\mathbb{M} \otimes_{ser}^{L} \mathcal{A}_\psi$ is reached, where the product $\otimes_{ser}^{L}$ is defined as follows:

Definition 4. *Consider two MDPs* $\mathbb{M}_1$ *and* $\mathbb{M}_2$. *For a given* ***connection*** $C : \mathbb{X}_1 \to \mathbb{U}_2$, *their* ***serial product*** *is the MDP* $\mathbb{M}_1 \otimes_{ser}^{C} \mathbb{M}_2 = (\mathbb{X}_1 \times \mathbb{X}_2, x_{ser}, \mathbb{U}_1, T_{ser})$, *where* $x_{ser} = (x_{1,0}, x_{2,0})$ *and* $T_{ser}(x_1', x_2' \mid x_1, x_2, u_1) = T_1(x_1' \mid x_1, u_1) T_2(x_2' \mid x_2, C(x_1'))$.

Additionally, we say that their ***parallel product*** *is the MDP* $\mathbb{M}_1 \otimes_{par} \mathbb{M}_2 = (\mathbb{X}_1 \times \mathbb{X}_2, x_{par}, \mathbb{U}_1 \times \mathbb{U}_2, T_{par})$, *where* $x_{par} = (x_{1,0}, x_{2,0})$ *and* $T_{par}(x_1', x_2' \mid x_1, x_2, u_1, u_2) = T_1(x_1' \mid x_1, u_1) T_2(x_2' \mid x_2, u_2)$.

Let us now return to the Mission Problem, where the MDP model contains both the rover MDP $\mathbb{M}_{rov}$ and the environment MDP $\mathbb{M}_{env}$. Since $\mathbb{M}_{env}$ is itself composed of several belief MDPs over individual labels, i.e., $\mathbb{M}_{env} = (\mathbb{M}_{e_1} \otimes_{par} \ldots \otimes_{par} \mathbb{M}_{e_n})$, it follows that the mission MDP is given as

$$\mathbb{M}_{miss} = \left(\mathbb{M}_{rov} \otimes_{ser}^{C_1} (\mathbb{M}_{e_1} \otimes_{par} \ldots \otimes_{par} \mathbb{M}_{e_n})\right), \tag{3}$$

where $C_1 : \mathbb{X}_{rov} \to \mathbb{U}_{e_1} \times \ldots \times \mathbb{U}_{e_n}$ is a measurement model that maps rover poses to measurements in the environment model. Moreover, when combined with the temporal specification φ we get an extended MDP composed of different parts: the rover, the environment belief model, and also the specification automaton $\mathcal{A}_\varphi$:

$$\mathbb{M}_1 = \left(\mathbb{M}_{rov} \otimes_{ser}^{C_1} (\mathbb{M}_{e_1} \otimes_{par} \ldots \otimes_{par} \mathbb{M}_{e_n})\right) \otimes_{ser}^{L} \mathcal{A}_\varphi. \tag{4}$$

Here $L : \mathbb{X}_{rov} \otimes \mathbb{X}_{env} \to \mathbb{U}_\varphi$ maps the joint space of rover poses and environment belief states to 2^{AP}, where AP is the set of atomic propositions used to construct φ. To design a policy for the mission phase, it is now sufficient to maximize the probability that the extended MDP (4) reaches the target set $\mathbb{X}_{rov} \otimes \mathbb{X}_{env} \otimes Q_f$.

Recall that the rover performs the mission task from time instant T_c (the end time of the copter exploration) until time $T_r + T_c$. Thus, for a given policy μ the probability of mission specification satisfaction (2) can be expressed as

$$\mathbb{P}_{\mu}^{\mathbb{M}_{miss}} \left[\mathbf{x}_{[T_c, T_c+T_r]} \models \varphi\right], \tag{5}$$

where $\mathbf{x}_{[T_c, T_c+T_r]}$ is the state trajectory over the interval $[T_c, T_c + T_r]$ of $\mathbb{M}_{miss}$. By incorporating the specification as part of the model, we can write an equivalent design problem

$$\arg\max_{\mu_1} \mathbb{P}_{\mu_1}^{\mathbb{M}_1} \left[\mathbf{x}^1_{[T_c, T_c+T_r]} \models \Diamond X_f\right] \tag{6}$$

with $\mathbf{x}^1_{[T_c, T_c+T_r]}$ being the state trajectory of the extended MDP $\mathbb{M}_1$ and with the target set $X_f = \mathbb{X}_{rov} \otimes \mathbb{X}_{env} \otimes Q_f$.

To solve the optimal reachability problem (6) we define a value function V_r that quantifies the probability that the rover satisfies its specification before time $T_r + T_c$ as a function of the system state $x_t := (x_t^r, x_t^e, x_t^\varphi)$ of $\mathbb{M}_1$ at time $t \geq T_c$:

$$V_r^t(x_t^r, x_t^e, x_t^\varphi) = \max_{\mu_1} \mathbb{P}_{\mu_1}^{\mathbb{M}_1} \left[\mathbf{x}^1_{[t, T_c+T_r]} \models \Diamond X_f \,|\, \mathbf{x}^1_{[t]} = x_t\right]. \tag{7}$$

From [1], we know that starting from $V^{T_c+T_r}(x) \equiv \mathbf{1}_{X_f}(x)$, this probability along with the optimal Markov policy can be inductively computed via *value iteration* as

$$V^t(x) = \max_{u \in \mathbb{U}_{rov}} \max\left(\mathbf{1}_{X_f}(x), \mathbb{E}\left[V^{t+1}(x') \mid x, u\right]\right), \tag{8}$$
$$\mu_1^t(x) \in \arg\max_{u \in \mathbb{U}_{rov}} \max\left(\mathbf{1}_{X_f}(x), \mathbb{E}\left[V^{t+1}(x') \mid x, u\right]\right),$$

for $x' \sim T(\cdot \,|\, x, u)$, and with $\mathbf{1}_{X_f}$ the indicator function of the set X_f.

Furthermore, for every Markov policy μ_1 for the extended MDP $\mathbb{M}_1$ there exists a policy μ for the rover-environment MDP $\mathbb{M}_{miss}$. But the nested products, e.g. in (4), lead to large aggregate state spaces, and since the size of a matrix representation of the transition kernel is in the worst case quadratic in the size of the state space, computation in the aggregate system quickly becomes challenging. In the next section we show that value iteration can be done recursively over subsystems without computing aggregate transition matrices, but first we discuss the exploration problem.

B. Optimal Control Solution of Exploration Problem

Similarly to above, we can model the overall stochastic system that is active in the exploration phase as

$$\mathbb{M}_2 = \mathbb{M}_{cop} \otimes_{ser}^{C_3} (\mathbb{M}_{e_1} \otimes_{par} \ldots \otimes_{par} \mathbb{M}_{e_n}), \tag{9}$$

with $C_3 : \mathbb{X}_{cop} \to \mathbb{U}_{e_1} \times \ldots \times \mathbb{U}_{e_n}$ being a measurement model that maps copter poses to measurements of the environment labels. The exploration phase affects the environment belief $x^e_{T_c}$ at time T_c. We assume knowledge of the initial rover state $x^r_{T_c}$ (the initial state $x^\varphi_{T_c}$ of the specification automaton is always known), and quantify the effect of the exploration as *mission risk* $R : \mathbb{X}_{env} \to [0, 1]$ defined as

$$R(x^e_{T_c}) = V_r^{T_c}(x^r_{T_c}, x^e_{T_c}, x^\varphi_{T_c}), \tag{10}$$

i.e., as a function of the environment belief state $x^e_{T_c}$ at the end of the exploration phase. Given $R(x^e_{T_c})$, the question now becomes how the environment should be explored in order to maximize the probability of a favorable $R(x^e_{T_c})$.

What would a good result look like? As a first consideration, one could choose the exploration objective

$$\max_{\mu_2} \ \mathbb{E}^{\mathbb{M}_2}_{\mu_2}(R(x^e_{T_c})), \tag{11}$$

which simply attempts to maximize the probability that the rover can satisfy the mission specification. We give a small example distilled from the larger Mars mission that highlights issues with this objective.

Example 2. *Consider an environment with a single belief state $x^e \in [0, 1]$ that expresses the belief that region A contains a sample of interest. We specify the science mission as "*get a sample of A*", i.e., $\Diamond$ sA, which is satisfied when A is reached and A contains a sample ($x^e = 1$). Without loss of generality, assume that the rover can reach A with probability 1 for some policy μ_R. Then μ_R is an optimal policy for the rover and the satisfaction probability is equal to the initial belief state x^e_0.*

Even if the copter can explore region A and thus determine exactly whether the mission can succeed or not, there is no benefit in doing so under the objective (11). Exploring A has an expected utility of $\mathbb{E}\left[x^e_{T_c}\right] = 1 \times x^e_0 + 0 \times (1 - x^e_0) = x^e_0$, i.e. identical to the initial value of (11). In other words, (11) does not encourage exploration.

Rather than naively maximizing the probability of satisfaction, which can discourage exploration, we also consider it a positive outcome if the exploration phase shows that the probability of completing the mission is very low. Based on such knowledge, resources can be redirected to more promising objectives. We can separate possible outcomes of the exploration phase into:

1) High confidence in mission completion, i.e.,
$$R(x^e_{T_c}) \geq 1 - \delta_{acc}, \tag{12}$$
where δ_{acc} is the maximal failure risk for which the rover mission can be accepted,
2) Very low confidence in mission completion, i.e.,
$$R(x^e_{T_c}) \leq \delta_{rej}, \tag{13}$$
where δ_{rej} is the maximal rejection risk for which the rover mission can be aborted,
3) Neither low nor high confidence in mission completion,
$$\delta_{rej} \leq R(x^e_{T_c}) \leq 1 - \delta_{acc}. \tag{14}$$

The third outcome is the least useful for decision-making purposes as it is associated with the largest degree of uncertainty about task feasibility.

Based on this classification, we consider the mission exploration a success if it results in $x^{T_c}_e$ being either in the "accept region" or in the "abort region". We can encode this condition as a subset of the copter-environment space

$$Y_\delta = \mathbb{X}_{cop} \otimes \{x^e_{T_c} \text{ s.t. (12) or s.t. (13)}\}, \tag{15}$$

and specify the exploration objective as

$$\max_{\mu_2} \mathbb{P}^{\mathbb{M}_2}_{\mu_2}\left[\mathbf{x}^2_{[0,T_c]} \models \Diamond Y_\delta \mid \mathbf{x}^2_{[0]} = (x^c_0, x^e_0)\right], \tag{16}$$

where we have used the symbol Y_δ as an atomic proposition for the associated target set. Again, this is a reachability objective, and the associated optimal probability and policy can be computed via the value iteration (8). Similar reachability objectives for uncertainty reduction have previously been proposed in the context of experiment design [14].

C. Overview

The solution developed in this section is summarized in the following algorithm:

Algorithm 1: Synthesize Exploration and Mission Plans

Plan Mission

1. Construct extended MDP $\mathbb{M}_1$ for specification φ;
2. Define target set $X_f = \mathbb{X}_{rov} \otimes \mathbb{X}_{env} \otimes Q_f$;
3. Solve the reachability problem (7) via value iteration to get mission policy μ_1 and mission risk $R(x^e_{T_c})$

Plan Exploration

4. Construct extended MDP $\mathbb{M}_2$ as in (9);
5. Define target set Y_δ based on $R(x^e_{T_c})$ and on $\delta_{acc}, \delta_{rej}$ as in eq. (15);
6. Solve the reachability problem (16) via value iteration to get exploration policy μ_2

IV. Value Iteration for Product MDPs

As detailed in the previous section, the aggregate systems $\mathbb{M}_1$ and $\mathbb{M}_2$ for the two phases are both formed as MDP products. Explicit construction of these aggregate systems quickly becomes computationally challenging. In this section, we show that explicit construction is not required for certain types of value iteration—it can be performed recursively over the product components.

Given a function $g : \mathbb{X} \times \mathbb{R}_+ \to \mathbb{R}_+$, consider a general Bellman operator[2] $\mathcal{B}$ on the following form:

$$(\mathcal{B}V)(x) = \max_{u \in \mathbb{U}} g\left(x, \mathbb{E}\left[V(x') \mid x, u\right]\right), \tag{17}$$

where g is a problem-specific function and the value function $V : \mathbb{X} \to \mathbb{R}_+$ is a positive mapping defined on the state space $\mathbb{X}$. For example, $g(x, v') = \max(\mathbf{1}_{X_f}(x), v')$ corresponds to the Bellman operator for reachability of a set X_f that we employed in Section III. Finite-horizon value iteration over a horizon $[0, T]$ consists of the following iterations:

$$\begin{aligned} V^T(x) &= g(x, 0), \quad V^{t-1} = \mathcal{B}V^t, \\ \mu^{t-1}(x) &\in \arg\max_{u \in \mathbb{U}} g\left(x, \mathbb{E}\left[V^t(x') \mid x, u\right]\right). \end{aligned} \tag{18}$$

We consider the Bellman iteration step for product MDPs. For a serial product $\mathbb{M} = \mathbb{M}_1 \otimes^C_{ser} \mathbb{M}_2$ we get

$$\begin{aligned} &\mathbb{E}^{\mathbb{M}}\left[V(x_1', x_2') \mid x_1, x_2, u_1\right] \\ &= \mathbb{E}^{\mathbb{M}_1}\left[\mathbb{E}^{\mathbb{M}_2}\left[V(x_1', x_2') \mid x_2, C(x_1')\right] \mid x_1, u_1\right], \end{aligned}$$

where we note that the inner expectation is a function of x_2 and x_1', but not of x_1. It follows that the Bellman update $V \mapsto \mathcal{B}V$ for a serial product can be computed in back-stepping fashion with an intermediate result W as follows:

$$\begin{aligned} W(x_1', x_2) &= \mathbb{E}^{\mathbb{M}_2}\left[V(x_1', x_2') \mid x_2, C(x_1')\right], \\ \mathcal{B}V(x_1, x_2) &= \max_{u_1 \in \mathbb{U}_1} g\left((x_1, x_2), \mathbb{E}^{\mathbb{M}_1}\left[W(x_1', x_2) \mid x_1, u_1\right]\right). \end{aligned}$$

Each step only requires knowledge of a single MDP component. Furthermore, the procedure immediately generalizes to n-length serial products where the $(n-i)$:th step becomes

$$\begin{aligned} &W_i(\ldots, x_{i-1}', x_i, x_{i+1} \ldots) \\ &= \mathbb{E}^{\mathbb{M}_i}\left[W_{i+1}(\ldots, x_{i-1}', x_i', x_{i+1} \ldots) \,|\, x_i, C_i(x_{i-1}')\right]. \end{aligned} \tag{19}$$

Remark 1. *In serial products of arbitrary length a connection C_i can be defined either as $C_i : \mathbb{X}_{i-1} \to \mathbb{U}_i$, or as $C_i : \prod_{j \in \Gamma_i} \mathbb{X}_j \to \mathbb{U}_i$ for some index set $\Gamma_i \subset \{1, \ldots, i-1\}$ that determines which preceding systems that affect the input to system i.*

In addition, if $\mathbb{M}_i$ is itself a serial or parallel product, the computation (19) can again be performed via recursive back-stepping. If for instance $\mathbb{M}_i = \mathbb{M}_{i_1} \otimes_{par} \mathbb{M}_{i_2}$ is a parallel product we can write $x_i = (x_{i_1}, x_{i_2})$ and (19) can be computed as

$$\begin{aligned} &W_{i_1}(\ldots (x_{i_1}', x_{i_2}) \ldots) \\ &\quad = \mathbb{E}^{\mathbb{M}_{i_2}}\left[W_{i+1}(\ldots (x_{i_1}', x_{i_2}') \ldots) \mid x_{i_2}, C_i(x_{i-1})\right], \\ &W_i(\ldots (x_{i_1}, x_{i_2}) \ldots) \\ &\quad = \mathbb{E}^{\mathbb{M}_{i_1}}\left[W_{i_1}(\ldots (x_{i_1}', x_{i_2}) \ldots) \mid x_{i_1}, C_i(x_{i-1})\right]. \end{aligned} \tag{20}$$

As a consequence, for any aggregate system constructed from a hierarchy of serial and parallel products, iterative value function computation for this type of Bellman operator can be performed without explicitly computing any aggregate transition probabilities. The back-stepping approach is advantageous when the connections are sparse, i.e., when the connection mappings C can be compactly represented.

Related Work. The back-stepping Bellman computation is an example of a sum-product algorithm [19]—a concept that has recently been used for computation of probabilistic invariance [11]. A generalization of serial and parallel products are *factored MDPs* [8]. In a factored MDP with local states $x_1, \ldots, x_n$ the state update for x_i depends only on the *scope* Γ_i, i.e. $T(x_i \mid x_1, \ldots, x_n) = T(x_i \mid x_j, j \in \Gamma_i)$. Evidently, this structure encompasses parallel and serial products: in a serial product $\Gamma_i = \{i-1, i\}$, whereas in a parallel product $\Gamma_i = \{i\}$. This structure has been exploited to obtain approximate methods for standard reward-based Bellman recursions [8, 13, 19]. Related ideas for value function approximation can potentially be used also within our framework to improve scalability further.

V. Case Study

We now apply our ideas in a case study. After introducing concrete models for rover, copter, environment belief, and measurements, we showcase the use of specification-guided exploration for different specifications and environments.

Abstract Robot Models. In practice, robotic systems operate in continuous domains, but value iteration is intractable over continuous state spaces. Methods to construct finite approximations of continuous models include abstraction-based [32] and sampling-based [2, 17] methods; an alternative is to employ approximate methods directly on the continuous state space [25]. For the purpose of clarity, we bypass these steps here and directly introduce finite-state models that capture the essential dynamics of agents that move around in a workspace. We denote the location of robot and copter with $x_r \in \mathbb{R}^2$ and $x_c \in \mathbb{R}^3$. For a given domain $\mathbb{X}_r \subset \mathbb{R}^2$ and $\mathbb{X}_c \subset \mathbb{R}^3$ we introduce abstract states $\xi_r \in [\mathbb{X}_r]_{N_{rx}, N_{ry}}$ and $\xi_c \in [\mathbb{X}_c]_{N_{cx}, N_{cy}, N_{cz}}$, where e.g. N_{rx} is the number of discrete states along the x axis.

We assume that low-level controllers are available for both robots so that they can move east, north, west, or south between the discrete states while remaining in the workspace, and that the copter can additionally adjust its elevation by moving up and down. With these assumptions, we can model the rover with an MDP $\mathbb{M}_{rov}$ with $N_{rx}N_{ry}$ states and 4 inputs, and the copter as an MDP $\mathbb{M}_{copt}$ with $N_{cx}N_{cy}N_{cz}$ states and 6 inputs. In the following we fix $N_{rx} = N_{ry} = N_{cx} = N_{cy} = 10$ and $N_{cz} = 2$, i.e. both robots move on a 10x10 grid in 2D space, and the copter can fly at two different altitudes (in the following referred to as *high* and *low*).

Environment Belief Model. We assume that regions of interest have been extracted from low-resolution satellite imagery, along with prior probability estimates for the likelihood that the regions exhibit certain traits. Here we restrict attention to *risk regions* that may contain obstacles that the rover can not traverse, and *target regions* that are likely places where scientific samples can be extracted. We associate to each region i

[2]For this set of Bellman recursions, the control input does not directly impact rewards—it only influences the value function via its effect on the next state distribution.

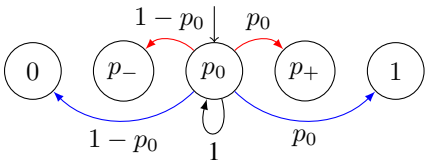

Fig. 5. Illustration of the environment belief model for a single region, only transitions from the initial state p_0 are shown. When a high-quality measurement is received (blue edges), the belief transitions to 1 with probability p_0 and to 0 with probability $1-p_0$, whereas a low-quality measurement yields beliefs p_- or p_+. If no measurement is taken (black) the state is unchanged.

a belief MDP $\mathbb{M}_{e_i}$ with five states $\zeta \in \{0, p_-, p_0, p_+, 1\}$ and three inputs $v \in \{NM, WM, SM\}$ for (N)o (M)easurement, (W)eak (M)easurement, and (S)trong (M)easurement. The transition matrices are $T_{NM} = I_5$,

$$T_{WM} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1-p_0 & 0 & p_0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, T_{SM} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1-p_- & 0 & 0 & 0 & p_- \\ 1-p_0 & 0 & 0 & 0 & p_0 \\ 1-p_+ & 0 & 0 & 0 & p_+ \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The outgoing transitions from p_0 are illustrated in Figure 5. Given belief MDPs $\mathbb{M}_{e_i}$ for every region of interest, we construct the environment model M_{env} as the parallel product

$$\mathbb{M}_{env} = \mathbb{M}_{e_1} \otimes_{par} \mathbb{M}_{e_1} \otimes_{par} \ldots \otimes_{par} \mathbb{M}_{e_n}.$$

Specification. The objective of the mission is to satisfy a scLTL specification φ over propositions over states in $\mathbb{M}_{rov}$ and $\mathbb{M}_{env}$. We consider two basic types of propositions:

- Do not be in a risk region R that may contain an obstacle: $\xi_r \notin R \vee \zeta_R = 0$.
- Collect a sample at region A: $\xi_r \in A \wedge \zeta_A = 1$.

We posit that the rover has a time window of length T_r to fulfill its mission, and that the copter has battery sufficient to operate for a time T_c. In addition, the copter must land in a predesignated landing area X_l at the end of the execution. This auxiliary objective is straightforward to incorporate into (16) by restricting the value iteration to policies that land safely with some high probability $1-\delta_l$.

Measurement Model. We connect the systems in serial as given by (4) and (9). The serial products are defined via connections given as follows:

- If the rover is adjacent to a region, it takes a (S)trong measurement of that region,
- If the copter is at low altitude and inside a region, it takes a (S)trong measurement of that region,
- If the copter is at high altitude and within distance 2 (infinity norm) of a region, it takes a (W)eak measurement of that region.

A. Scenarios

We demonstrate the features of the proposed solution in a few example scenarios. In all scenarios we assume that a weak measurement has accuracy 0.85, i.e. that a positive weak measurement yields a belief state $\zeta = 0.85$. We limit the rover to $T_r = 15$ steps and the copter to $T_c = 30$ steps. Since the rover moves much slower than the copter these two time intervals are not directly comparable: the copter will complete its 30 steps much faster than the rover completes 15 steps. In other words, the sampling times are different. Furthermore we set $\delta_{acc} = \delta_{rej} = 0.1$ which implies that the goal of the exploration phase is to show that the mission can be completed with at least 90% probability, or that the success probability is at most 10%. For these examples, policy synthesis for both the mission and exploration phases takes around 60 seconds on a 2GHz laptop, using our prototype Python implementation.

Scenario 1. The left part of Fig. 6 illustrates the Kimberly region in the Mars Gale Crater where regions have been labeled as obstacles (red), risk regions (orange) and science target regions (blue). While the obstacle regions $R_1 - R_3$ have been determined to be non-traversable, there is a rock area R_4 and a sandy area R_5 that may be possible to cross. The objective is to collect a sample from any one of the three target regions while avoiding obstacles, i.e. the specification is $\varphi = \neg\texttt{obstacle}\ \mathcal{U}\ \texttt{sA}$, where the atomic proposition `obstacle` is true if the rover is in a risk region that contains an obstacle, and `sA` is true if the rover is in a target region that contains a science sample.

We consider a scenario where the initial probability of specification satisfaction is high: there is a high estimated probability that a sample can be extracted from the easy-to-access region A_3. However, when the rover reaches A_3 it turns out to be empty and the probability of mission failure increases. At this moment additional exploration is warranted to reduce the mission uncertainty, so we apply specification-guided exploration to decide on the continuation of the mission.

With the two-stage approach described in Section III we can compute 1) a mission policy for the rover and the associated risk function $R(\xi^e_{T_c})$, and 2) a policy for the copter that maximizes the probability of reaching a state where $R(\xi^e_{T_c})$ is either close to 0 or close to 1. In Fig. 7 the resulting team behavior is illustrated for two configurations of the environment: one where the specification can be satisfied and one where it is infeasible and the mission is aborted. There is no need to re-compute a mission policy after exploration—only the environment belief state is changed, and the mission policy is optimal for all belief states. The following video shows excerpts from a simulation of the successful exploration and mission: https://youtu.be/ZKKFOiXSeaw.

Scenario 2. To further illustrate exploration and cooperative behavior, we consider the artificial example depicted in the right part of Fig. 6. The specification is as follows: either collect samples of both types A and B, or collect a sample of type C, while avoiding obstacles. Letting $\Diamond_o\varphi = \neg\texttt{obstacle}\ \mathcal{U}\ \varphi$, the specification can be written $(\Diamond_o\texttt{sA} \wedge \Diamond_o\texttt{sB}) \vee (\Diamond_o\texttt{sC})$. The atomic proposition `sA` is true if $\xi_r \in A$ (rover is in region A) and $\zeta_A = 1$ (sample detected in A), and similarly for `sB` and `sC`.

Executions generated from the resulting copter and rover policies are shown in Fig. 8 for different configurations of the true environment. With only prior knowledge about the

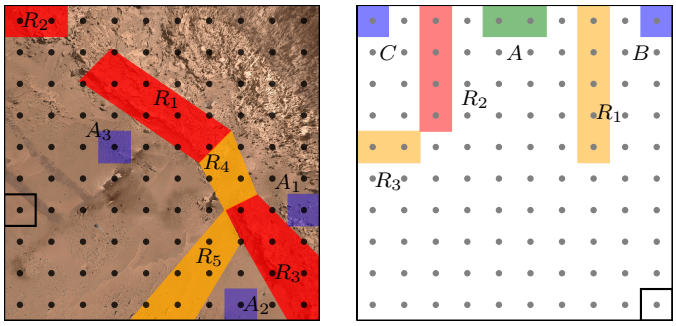

Fig. 6. Views of the two example work spaces where regions of interest are marked. In the left scenario the prior probabilities for A_1, A_2, A_3, R_4, and R_5 are 0.5, 0.5 0.9, 0.4 and 0.3, respectively. In the right scenario the prior probabilities are 0.5 for all regions except R_2 which is a known obstacle. Each abstract state is illustrated with a dot. The black square marks the required landing zone for the copter.

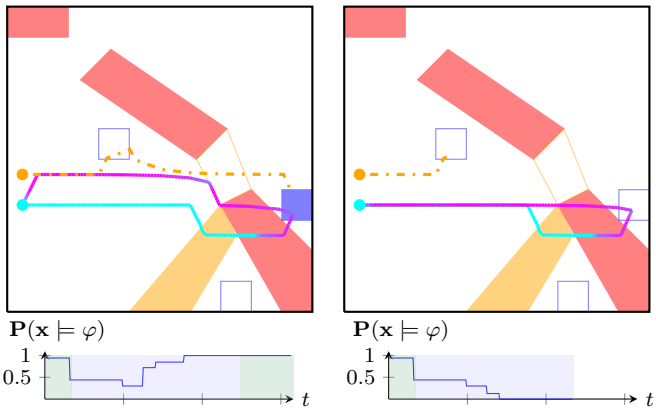

Fig. 7. Above: Trajectories of the rover (orange dash-dot) and copter (solid teal/purple for low/high elevation), for two environment configurations. Non-filled regions indicate that the true state is equal to 0, i.e. there is no sample or obstacle. Below: Estimated probability of specification satisfaction over time. In the initial mission phase (green), the rover moves towards a target area that has a high sample probability of 0.9. However, the target area unexpectedly turns out to not contain a sample and the probability of specification satisfaction decreases to around 0.5. This triggers an exploration phase (blue) where the copter attempts to find out whether the mission can be completed or not. This succeeds in the illustration to the left where A_1 contains a sample and the rover proceeds with its mission, but fails in the case to the right where A_1 is empty and the mission is aborted.

environment, the probability that the rover can satisfy the specification is $R(\xi_0^e) = 0.25$. However, after the copter exploration significant amounts of uncertainty are removed: for the two upper feasible examples in Figure. 8, $R(\xi_{T_c}^e) = 1$ and 0.85 respectively, where $\xi_e^{T_c}$ is the environment belief state after copter exploration, and $R(\xi_{T_c}^e) = 0.075$ for the lower infeasible examples. This shows that the generated policy explores the environment in a way that efficiently reduces uncertainty about whether the specification can be satisfied. For comparison also the optimal behavior *without* copter exploration is illustrated; it turns out that the optimal behavior is to bet that there is a reachable sample in region C. This succeeds only in the upper right scenario (the trajectories overlap). In the upper left scenario, the guess is wrong and in the lower scenarios rover locomotion time is wasted trying to complete an impossible mission.

In conclusion, both scenarios showcase the benefit of exploration, and in particular that *specification-guided* exploration

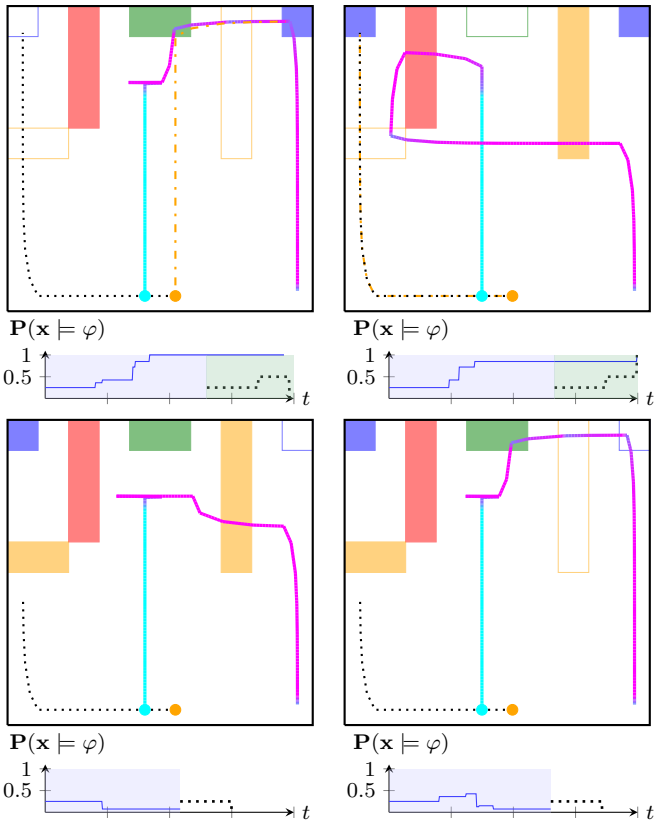

Fig. 8. Illustration of the copter and rover policies for different environment configurations in the artificial scenario. Interpretation is equivalent to that of Fig. 7. All four executions are generated by the exact same policies; they adapt according to what is encountered in the environment. In the upper two examples the exploration phase shows high likelihood of specification satisfaction, whereas in the lower examples the specification is found to likely be infeasible and the mission is aborted. For comparison, also the corresponding trajectory and probability *without exploration* are plotted with black dotted lines.

can efficiently collect information required to decide whether a mission should be continued or aborted.

VI. Conclusions and Future Work

We have presented a modeling framework for cooperative robot teams operating in uncertain environments. Within this framework we have solved the environment exploration problem in a specification-dependent manner via a dynamic programming-based solution. This specification-dependent solution allows for targeted exploration of large unknown environments. We have shown that the curse of dimensionality in value iteration can be partially mitigated by using only implicit representations of the aggregate systems, but this does not completely solve the scalability issue: an exact representation of the value function is invariably proportional to the size of the aggregate state space, which in turn grows exponentially with the number of system components.

For future work, we are interested in improving scalability further by using approximate value function representations, and in incorporating more complicated environment models for which more knowledge about one region also says something about neighboring or similar regions.

References

[1] A. Abate, S. S. Sastry, M. Prandini, and J. Lygeros. Probabilistic Reachability and Safety for Controlled Discrete Time Stochastic Hybrid Systems. *Automatica*, 44(11): 2724–2734, 2008.

[2] A. Agha-mohammadi, S. Chakravorty, and N. M. Amato. FIRM: Sampling-based feedback motion-planning under motion uncertainty and imperfect measurements. *The International Journal of Robotics Research*, 33(2):268–304, 2014.

[3] A. Agha-mohammadi, E. Heiden, K. Hausman, and G. Sukhatme. Confidence-rich grid mapping. In *International Symposium on Robotic Research*, 2017.

[4] J. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active vision. *International Journal of Computer Vision*, 1(4): 333–356, 1988.

[5] C. Belta, B. Yordanov, and E. A. Gol. *Formal Methods for Discrete-Time Dynamical Systems*. Springer, 2017.

[6] D. P. Bertsekas and S. E. Shreve. *Stochastic Optimal Control: The Discrete Time Case*. Academic Press, 1978.

[7] A. Blake and A. Yuille. *Active vision*. MIT Press, 1992.

[8] C. Boutilier, R. Dearden, and M. Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1-2):49–107, 2000.

[9] E. Cristofalo, K. Leahy, C. I. Vasile, E. Montijano, M. Schwager, and C. Belta. Localization of a Ground Robot by Aerial Robots for GPS-deprived Control with Temporal Logic Constraints. In *Proc. ISER*, pages 525–537, 2016.

[10] A. J. Davison and D. W. Murray. Simultaneous localization and map-building using active vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 (7):865–880, 2002.

[11] S. Esmaeil Zadeh Soudjani, A. Abate, and R. Majumdar. Dynamic Bayesian networks for formal verification of structured stochastic processes. *Acta Informatica*, 54(2): 217–242, 2017.

[12] H. J. S. Feder, J. J. Leonard, and C. M. Smith. Adaptive mobile robot navigation and mapping. *International Journal of Robotics Research*, 18(7):650–668, 1999.

[13] C. Guestrin, D. Koller, R. Parr, and S. Venkataraman. Efficient solution algorithms for factored MDPs. *Journal of Artificial Intelligence Research*, 19(c):399–468, 2003.

[14] S. Haesaert, P. M. Van den Hof, and A. Abate. Experiment design for formal verification via stochastic optimal control. In *Proc. ECC*, pages 427–432, 2016.

[15] A. Jones, M. Schwager, and C. Belta. Distribution temporal logic: Combining correctness with quality of

[17] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on* estimation. In *Proc. IEEE CDC*, pages 4719–4724, 2013.

[16] S. Karaman and E. Frazzoli. Sampling-based Optimal Motion Planning with Deterministic μ-Calculus Specifications. In *Proc. ACC*, 2012.
Robotics and Automation, 12(4):566–580, 1996.

[18] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas. Where's Waldo? Sensor-based temporal logic motion planning. In *Proc. IEEE ICRA*, pages 3116–3121, 2007.

[19] F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

[20] M. Lahijanian, S. B. Andersson, and C. Belta. Temporal logic motion planning and control with probabilistic satisfaction guarantees. *IEEE Transactions on Robotics*, 28(2):396–409, 2012.

[21] K. Leahy, A. Jones, M. Schwager, and C. Belta. Distributed information gathering policies under temporal logic constraints. In *Proc. IEEE CDC*, pages 6803–6808, 2015.

[22] S. C. Livingston and R. M. Murray. Just-in-time synthesis for motion planning with temporal logic. In *Proc. IEEE ICRA*, 2013.

[23] M. 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 *Proc. HSCC*, pages 353–362, 2013.

[24] B. Mu, M. Giamou, L. Paull, A. Agha-mohammadi, J. J. Leonard, and J. P. How. Information-based active SLAM via topological feature graphs. In *Proc. IEEE CDC*.

[25] W. B. Powell. *Approximate Dynamic Programming*. Wiley, 2011.

[26] S. Soatto. Actionable information in vision. In *Machine learning for computer vision*, pages 17–48. Springer, 2013.

[27] C. Stachniss. *Robotic mapping and exploration*. Springer, 2009.

[28] M. Svorenova, I. Cerna, and C. Belta. Optimal control of mdps with temporal logic constraints. In *Proc. IEEE CDC*, pages 3938–3943, 2013.

[29] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[30] S. Thrun et al. Robotic mapping: A survey. In *Exploring artificial intelligence in the new millennium*, pages 1–35. 2002.

[31] C. I. Vasile and C. Belta. Sampling-Based Temporal Logic Path Planning. In *Proc. IEEE/RSJ IROS*, 2013.

[32] M. Zamani, A. Abate, and A. Girard. Symbolic models for stochastic switched systems: A discretization and a discretization-free approach. *Automatica*, 55:183–196, 2015.