

Safe Mission Planning under Dynamical Uncertainties

Yimeng Lu and Maryam Kamgarpour*

Abstract

This paper considers safe robot mission planning in uncertain dynamical environments. This problem arises in applications such as surveillance, emergency rescue, and autonomous driving. It is a challenging problem due to modeling and integrating dynamical uncertainties into a safe planning framework, and finding a solution in a computationally tractable way. In this work, we first develop a probabilistic model for dynamical uncertainties. Then, we provide a framework to generate a path that maximizes safety for complex missions by incorporating the uncertainty model. We also devise a Monte Carlo method to obtain a safe path efficiently. Finally, we evaluate the performance of our approach and compare it to potential alternatives in several case studies.

1 Introduction

With the advances of robotics and artificial intelligence, autonomous robots are increasingly used in safety-critical applications. These applications, such as surveillance [11], emergency rescue [38], and autonomous driving [9], all require the robot to plan its path under uncertainties in the environment as well as in the model and motion of other robots. Path planning in these cases is challenging for the following reasons. First, it is required that the generated path has the highest probability of remaining safe while completing the task. Second, these uncertainties are in general dynamical and originate from different sources, e.g., uncertain locations of the obstacles [39], targets [17], and/or an evolving hazard [38]. Modeling these uncertainties in a computationally tractable way and incorporating such models into a safe planning framework is a central challenge in applying robots in real-world scenarios. Last, complex missions usually involve multiple stages where several subtasks need to be fulfilled. The generated plan should handle both low-level point-to-point planning for each subtask and high-level decision making of execution order. In a search and rescue scenario, for example, the robot needs to find a path to visit all the critical locations under an evolving hazard while maximizing its safety. To address the challenges above, this paper aims at providing a framework for planning with safety guarantees in uncertain dynamical environments.

In most applications of practical interest, uncertainties such as the locations of obstacles, pedestrians, vehicles, and hazardous areas, change with time. However, many of the current approaches for planning assume that these external factors are static in their initial planning stage [2, 12, 16, 23]. To mitigate this issue, reactive replanning methods such as D* [32], D* Lite [19] and RRT^X [26] adapt the initially generated path when a potential collision is detected during execution. Nonetheless, these methods are not safety maximizing as they do not account for uncertainty model during

*This research was gratefully funded by the Swiss National Science Foundation, under the grant SNSF 200021_172781 and the ETH Zurich Research Grant. The authors are with the Automatic Control Laboratory, Department of Information Technology and Electrical Engineering, ETH Zürich, Switzerland. E-mails: {luyi, mkamgar}@control.ee.ethz.ch

This paper appears in International Conference of Robotics and Automation (ICRA 2020), Paris, France.

planning. Reactively adapting the path might not be globally optimal with respect to the safety of the whole mission. There also exist approaches to account for the uncertainty model such as [13,27], where a known deterministic dynamic model of uncertainty is assumed for optimal safe planning. However, it is generally not realistic to assume perfect knowledge of the uncertainty dynamics in practical applications such as emergency rescue and autonomous driving. Chance-constrained RRT in [24] assumes a Gaussian model for dynamic obstacles and computes a probabilistically guaranteed feasible path offline. Its extension in [1] applies chance-constrained RRT to predicted future behaviors of the dynamical obstacles. Modeling the uncertainties as a partially observable Markov decision process (POMDP) is investigated in [39] and [5] in order to obtain a collision-free path at a crossroad with dynamic programming (DP). However, these approaches restrict uncertainties to Gaussian distributions [1, 24, 39], or generalize the uncertainties to Markov motion models with potentially intractable state-space [5].

In scenarios with complex specifications, model-checking tools such as temporal logic verification can be used to handle both high-level decision making and low-level planning [20, 22, 36]. There also exist works in temporal logic planning with uncertainties. In [15, 25, 28, 37], a preliminary plan is derived first and then revised reactively when the original plan is found infeasible during execution. However, similar to [2, 12, 16, 23], these works do not maximize overall safety as they do not use a model of uncertainty dynamics. Incorporating uncertainty dynamics into planning frameworks increases the state space and thus, also the computational complexity [38]. Other approaches include learning the uncertainty dynamics [7, 9] and incremental synthesis for temporal logic specifications [35]. However, their uncertainty models are for either specific scenarios or limited dimensions, which restricts the applicability for more general situations.

As an example of addressing dynamical uncertainties in safety-critical applications of robotics, we consider an emergency rescue mission in an environment affected by a spreading hazard or contamination such as fire or toxic gas. We consider designing paths for the first responders or robots to maximize their safety. Using robots in safety-critical tasks is receiving increasing attention in robotics community [6, 18, 21]. For the problem mentioned above, the uncertain hazard is dynamical, and its evolution cannot be precisely known. Developing and incorporating uncertainty models and finding computationally tractable approaches are challenging for these applications. Models and simulations of the hazardous environment are discussed in [8, 31], but these detailed models are mainly used for verification rather than stochastic control design. Recent works in emergency rescue are constrained to simple environmental settings. For instance, the works in [3, 4] discussed robot planning in earthquakes, but the models used are static. Robot-assisted evacuation is considered in [14, 29, 30], but dynamical uncertainty models are not incorporated. In [38], a similar search and rescue mission was considered for the case of firefighting in a building affected by fire, and a solution approach based on [17] was introduced. While the formulation of [17] allows for general environmental uncertainties, the computational approximations introduced there for tractability might result in poor safety performance.

Our contributions are summarized as follows. First, we represent the evolution of the dynamical uncertainties using a probabilistic model. Second, we develop an approximate dynamic programming algorithm to incorporate this model of uncertainties into the planning problem and solve it in a computationally tractable way; this is achieved by approximating the uncertainty evolution using a Monte Carlo method and providing corresponding safety guarantees. Third, we compare our approach with past works such as D* Lite in several case studies to show that our method provides safer path planning under dynamical uncertainties.

The rest of the paper is organized as follows. Section II formulates planning problems in uncertain dynamical environments. In Section III, we discuss our approach to handle dynamical uncertainties and solution to the planning problem in a tractable manner. In Section IV, we provide

case studies for illustrating the approach and comparing it with potential alternatives. Section V concludes the paper and discusses future works.

2 System model and problem formulation

We consider a general planning framework for an agent in an uncertain dynamical environment. We first define the models of the robot and uncertainties. Then, we define the planning problem for complex specifications, followed by the control synthesis of the problem.

2.1 System model

2.1.1 Grid space and robot dynamics

We use a discrete 2-dimensional grid $X = \{0, 1, \dots, m-1\} \times \{0, 1, \dots, n-1\}$ to indicate different locations in the environment. A grid cell can be free space or occupied by obstacles. A map is a function $\mathcal{M} : X \rightarrow \{0, 1\}$, whose value is 0 for free space and 1 for an obstacle such as a wall. We use the Manhattan distance, which is the sum of the horizontal and vertical distances between cells, and assume that each grid cell has size 1×1 .

We assume that an agent located at $x = (x^1, x^2) \in X$ can move in free space $X_f := \{x \in X \mid \mathcal{M}(x) = 0\}$. Let $N(x)$ denote the subset of X_f reachable from x in one step using control input $u \in U := \{N, S, E, W, 0\}$, denoting the four directions of movement as well as staying in the current position. Given $x_t \in X_f$, the location at the next time step x_{t+1} follows a distribution $x_{t+1} \sim \tau^X(\cdot \mid x_t, u_t)$, where $x_{t+1} \in N(x_t)$, $u_t \in U$.

2.1.2 Uncertain dynamics of the environment

To model arbitrary uncertainties with Markov dynamics, we define the parameterized stochastic set process below.

Definition 1 (Stochastic set process) *Let $Y := \{0, 1\}^{m \times n}$ be the set of all binary matrices with the same dimension as the grid X . The transition probability between each $y \in Y$ is determined by a stochastic kernel $\tau^Y : Y \times Y \rightarrow [0, 1]$. A stochastic set process is defined by the Markov process $y_{t+1} \sim \tau^Y(\cdot \mid y_t)$, $t \in \{0, 1, \dots\}$ with its initial condition y_0 , and a set valued map $\gamma : Y \rightarrow X$. In our case, $\gamma(y) = \{x \in X \mid [y]_x = 1\}$, where $[y]_x$ is the element at x position of the binary matrix y .*

We highlight that the above stochastic set framework allows for any uncertainty with Markov dynamics. For example, parameterized geometric shapes such as ellipses or rectangles arise from uncertain vehicle motions [33] and probabilistic hazards. The Markov parameter y can denote the center of the shape, and the map γ can denote the volume. Note that our setup is more general as it allows for arbitrarily shaped obstacles or hazards. Consequently, it becomes more computationally challenging, and hence, it is essential to develop a method to address this problem.

2.2 Problem formulation

We use an evolving hazard as an example of dynamical uncertainties in the following discussions. The mission is to visit several targets and then reach the final goal location while keeping the robot out of the hazardous areas, which arises in search and rescue scenarios. For example, in Figure 1, the mission can be accomplished by starting from the initial location to visit targets A and B in an unspecified order, and exit at goal location F. And the hazard $H = H_1 \cup H_2 \cup H_3$ is

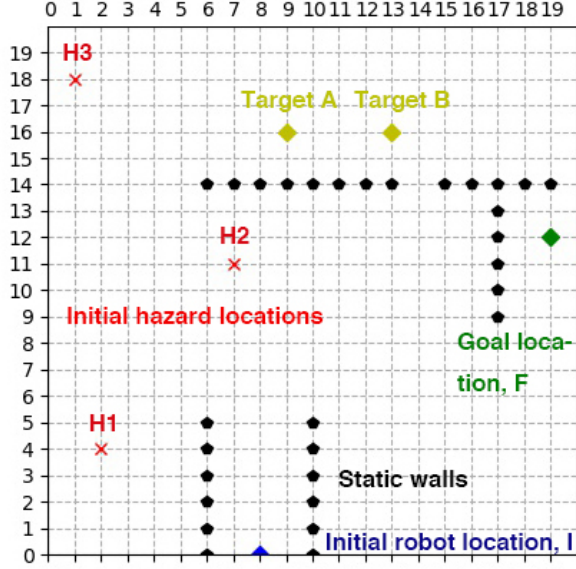


Figure 1: An example of a complex mission under uncertainties

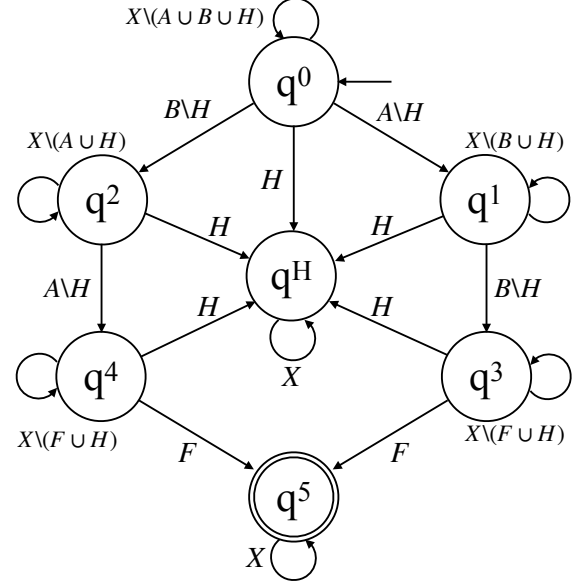


Figure 2: Automaton example of the setup in Fig. 1

shown in its initial condition. This complex mission can be described as a logical combination of visiting different targets while remaining safe and be modeled by a deterministic finite automaton (DFA) [10]. We form an overall system by combining the DFA and system dynamics. We present optimal planning as an automaton specification satisfaction problem. Note that due to the presence of the uncertainties, our objective becomes maximizing the probability of safely completing the desired task with the generated path.

2.2.1 Automaton specification satisfaction

Let us formally define the finite state automaton to describe the considered specifications and the solution approach systematically.

Definition 2 A finite-state automaton is a tuple $\mathcal{A} = (Q, Q^I, Q^F, \Sigma, \Delta)$, where Q is a finite set of $n_q \in \mathbb{N}$ states, $Q^I \subset Q$ is a set of initial states, $Q^F \subset Q$ is a set of final (goal) states, Σ is a finite alphabet, and $\Delta \subset Q \times \Sigma \times Q$ is a transition relation.

Let the parameterized set $\gamma^i : Y \rightarrow X$ determine the transition to automaton state i . The automaton alphabet is defined as $\Sigma := 2^A$ and its transition relation is $\Delta := \{(j, \sigma, i) | \sigma = \{K^i\}, \forall i \in Q\}$, where $K^i := \{(x, y) | x \in \gamma^i(y)\}$ and $A := \cup_{i=1}^{|Q|} K^i$ are defined in the product space $X \times Y$. For any final state $q \in Q^F$, we define $(q, X \times Y, q) \in \Delta$ making each final automaton state absorbing. This ensures once the final state is reached, the specification is satisfied as will be discussed below. We denote the hazard automaton state as q^H and following the transition $(q^H, X \times Y, q^H) \in \Delta$ to make q^H absorbing. This ensures once the robot enters a hazard state, the trajectory is no longer considered safe. With the automaton, we can define the specification as follows.

Definition 3 (Specification) A robot trajectory $\{x_t\}_{t=0}^N$ satisfies a specification given by \mathcal{A} , if there exists a sequence of automaton states (q_0, q_1, \dots, q_N) , such that $q_0 \in Q^I$, $(q_{t-1}, (x_t, y_t), q_t) \in \Delta$ for $t = 1, 2, \dots, N$, and $q_N \in Q^F$.

We denote the resulting specification encoding automaton by \mathcal{A}_s . We make the assumption that the set $\{K^i\}_{i=1}^{|Q|}$ partitions $X \times Y$. Under this assumption \mathcal{A}_s is deterministic and non-blocking [38], thus ensuring that the following transition kernel on $Q \times X \times Y$ to be well-defined. We denote the deterministic automaton transition kernel by $\tau^Q : Q \times Q \times X \times Y \rightarrow [0, 1]$ and $\tau^Q(q'|q, x', y') = 1_{\bar{q}'(q')}$, where $(q, (x', y'), \bar{q}') \in \Delta$. Then, we combine the above mentioned states to form the overall system state $s \in S := Q \times X \times Y$. The combined stochastic transition kernel between system states is $\tau(q', x', y'|q, x, y, u) = \tau^Q(q'|q, x', y') \tau^X(x'|x, u) \tau^Y(y'|y)$. This kernel provides the probability of transiting to a state $(q', x', y') \in S$ given the state $(q, x, y) \in S$ and control input $u \in U$. For path planning in hazardous environments, the automaton is a high-level state machine indicating the stage of the mission and the status of the system. The automaton for the mission in Figure 1 is shown in Figure 2.

2.2.2 Control synthesis and dynamic programming solution

Let the goal set be denoted by $G \subset S$ as $G := Q^F \times X \times Y$. It follows that a trajectory $\{x_t\}_{t=0}^N$ satisfies a specification given by automaton if and only if $\exists t \in \{0, 1, \dots, N\}$ such that $s_t \in G$. As the environment is stochastic, we are concerned with the probability of a trajectory $\{x_t\}_{t=0}^N$ satisfying the specification given a series of control policy $\mu := \{\mu_0, \mu_1, \dots, \mu_{N-1}\}$, where $\mu_t : S \rightarrow U$. This probability is defined as

$$r_N(s_0, \mu) := \Pr_{s_0}^\mu \left(\bigcup_{t=0}^N (S \setminus G)^t \times G \times S^{N-t} \right),$$

which is maximized by an optimal μ . The event $(S \setminus G)^t \times G \times S^{N-t}$ indicates that the goal set G is reached exactly at t , i.e., $s_t \in G \wedge s_k \notin G, \forall k < t$.

A backward recursion can be derived to solve for $r_N^*(s_0)$ above, with $V_N^*(q, x, y) = 1_G(q, x, y)$ and for $1 \leq t \leq N$:

$$V_{t-1}^*(q, x, y) = 1_G(q, x, y) + 1_{S \setminus G}(q, x, y) \max_{u \in U} \sum_{q' \in Q} \sum_{x' \in X} \sum_{y' \in Y} V_t^*(q', x', y') \tau^Q(q'|q, x', y') \tau^X(x'|x, u) \tau^Y(y'|y). \quad (1)$$

where the indicator function $1_G(s) = 1$ if $s \in G$.

It is known that that $r_N^*(s_0) = V_0^*(s_0)$ and the corresponding optimal control inputs obtained from the backward recursion give us the optimal control policies $\mu_t(s)$ for the satisfaction problem [34]. However, this backward recursion needs to be solved over the entire state space $S = Q \times X \times Y$, and the size of the stochastic environment state space Y is exponential in the size of the grid space X . In the following section, we present our approach to mitigate this intractability issue by decoupling the environment state from the overall state space and exploit a Monte Carlo approximation for computational efficiency.

3 Using safe transition probability to handle dynamical uncertainties

In order to decouple the environment state from the backward recursion, we first propose the notion of safe transition probability to model the evolution of the hazard process using its initial condition only. Then, to alleviate the computation complexity, we approximate the safe transition probability

with a Monte Carlo approach. Using the above two techniques, we solve a computationally tractable version of (1) without Y as part of the state space.

3.1 Modeling uncertainties using safe transition probability

To introduce our novel notion **safe transition probability (STP)**, we first recognize the redundancy and coupling in the system model from the following lemmas.

Lemma 1 (Redundancy of the transition dynamics) *The automaton transition kernel $\tau^Q(q'|q, x', y')$ can be determined by q , x' and $[y']_{x'}$, i.e., only one element $[y']_{x'}$ of the binary matrix y' is needed rather than the whole matrix.*

Proof *From the definition of τ^Q , the value of $\tau^Q(q'|q, x', y')$ is 1 if $(q, (x', y'), q') \in \Delta$ and 0 otherwise. The transition from q to q' occurs if $x' \in \gamma^{q'}$. We conclude only $[y']_{x'}$ is used in y' .* ■

When computing $V_{t-1}^*(q, x, y)$ from (1), for $q = q^H$, the associated value functions are 0, i.e., $V_{t-1}^*(q^H, x, y) = 0$ for $\forall t, x$ and y . For $q \neq q^H$, $V_{t-1}^*(q, x, y) \neq 0$ is true only for the environment state y such that $[y]_x = 0$, following **Lemma 1**. This holds because in order to have non-zero value function, a state should be able to reach the goal set G . We conclude this with **Lemma 2**.

Lemma 2 (Coupling of the state elements) *Any state such that $V_t^*(q, x, y) \neq 0$ must have $q \neq q^H$ and $[y]_x = 0$.*

Proof *If $[y]_x = 1$, we will have $q = q^H$ according to the transition dynamics. Then, $V_t^*(q, x, y) = 0$ because this state can never reach G from (q, x, y) at t . We conclude the lemma by the contraposition of this statement.* ■

In order to decouple the environment state y while solving (1) at $t - 1$, we need to account for the coupling between q and $[y]_x$ stated in **Lemma 2**. We define the **safe transition probability (STP)** of the environment state being $y_t = y'$, with the knowledge of y_0 , and conditioned on q and x as:

$$\tau_t^Y(y'; q, x) = \begin{cases} \sum_{\substack{y_{t-1} \in Y \\ [y_{t-1}]_x = 0}} \tau^Y(y'|y_{t-1}) \sum_{y_{t-2} \in Y} \cdots \sum_{y_1 \in Y} \prod_{k=1}^{t-1} \tau^Y(y_k|y_{k-1}), & q \neq q^H \\ 0, & q = q^H \end{cases} \quad (2)$$

for $t \geq 1$ and $\tau_0^Y(y') = 1_{y_0}(y')$.

Intuitively, STP is another way of expressing probability of the environment state transiting to y' at t , which was expressed as $\tau^Y(y'|y)$ in (1). As we will decouple the environment state, we will not be able to access y , but use q and x to infer this probability from STP.

3.2 Incorporating STP in the backward recursion framework

Our first result is that we can remove the environment state y from the backward recursion state space using STP.

Theorem 1 (Independence of hazard process and optimal solution) *By modeling the environment state evolution using STP, the value function and the optimal control policy obtained are independent of the parameter space Y .*

Proof We omit x, y in $V_N^*(s)$, $1_G(s)$ and $1_{S \setminus G}(s)$ as they do not depend on x and y . We obtain the approximated optimal value function $\tilde{V}_{N-1}^*(q, x, y)$ for any state $s = (q, x, y)$ using STP as

$$\begin{aligned} \tilde{V}_{N-1}^*(q, x, y) &= 1_G(q) + 1_{S \setminus G}(q) \max_{u \in U} \sum_{q' \in Q} \sum_{x' \in X} \sum_{y' \in Y} \\ &\quad V_N^*(q') \tau^Q(q'|q, x', y') \tau^X(x'|x, u) \tau_N^Y(y'; q, x) \\ &= 1_G(q) + 1_{S \setminus G}(q) \max_{u \in U} \sum_{q' \in Q} \sum_{x' \in X} V_N^*(q') \tau^X(x'|x, u) \\ &\quad \sum_{y' \in Y} \tau^Q(q'|q, x', y') \tau_N^Y(y'; q, x). \end{aligned}$$

In the first equality, we replace the environment dynamics $\tau^Y(y'|y)$ with STP at $t = N$, as STP also represents the probability of $y_N = y'$ but with the knowledge of y_0 only. From the second equality, it follows that the value function $\tilde{V}_{N-1}^*(\cdot)$ is only a function of q and x . We use induction to conclude that $\tilde{V}_t^*(\cdot)$ is independent of the parameter $y \in Y$ for $t = N - 1, \dots, 0$ by replacing $V_{t+1}^*(q', x', y')$ by $\tilde{V}_{t+1}^*(q', x')$ in the backward recursion. ■

We use the notation $\tilde{V}_t^*(q, x)$ in the simplified dynamic programming with STP. Note that $\tilde{V}_t^*(q, x)$ approximates $V_t(q, x, y)$ by only considering two out of three arguments, because we no longer have access to the environment state which is eliminated in the summation over y' . We rewrite (1) using STP as

$$\begin{aligned} \tilde{V}_{t-1}^*(q, x) &= 1_G(q) + 1_{S \setminus G}(q) \max_{u \in U} \sum_{q' \in Q} \sum_{x' \in X} \tilde{V}_t^*(q', x') \\ &\quad \tau^X(x'|x, u) \sum_{y' \in Y} \tau^Q(q'|q, x', y') \tau_t^Y(y'; q, x). \end{aligned} \quad (3)$$

Note that when we compute $V_{t-1}^*(q, x, y)$ from (1), those states associated with y such that $[y]_x = 1$ have zero value functions (**Lemma 2**). Thus, y' such that $[y]_x = 1$ in the summation in (1) are effectively excluded for the computation of $V_{t-1}^*(q, x, y)$. When computing $\tilde{V}_{t-1}^*(q, x)$ in (3), only those y' that satisfy $[y]_x = 0$ should be considered in the summation over y' for the same reason. Also, $\tilde{V}_{t-1}^*(q^H, x) = 0$ should be true for $\forall t$ and x , following the original full state solution. We use the following theorem that the solution of (3) also complies with this.

Theorem 2 *By using STP, the solution $\tilde{V}_{t-1}^*(q, x)$ obtained from (3) satisfies: (a) For $q \neq q^H$, only y' such that $[y]_x = 0$ are included in the summation in backward recursion (3). (b) For $q = q^H$, $\tilde{V}_{t-1}^*(q^H, x) = 0$ for $\forall t$ and x .*

Proof While computing $\tilde{V}_{t-1}^*(q, x)$ when $q \neq q^H$, all the y' in the summation satisfies $[y]_x = 0$ by the definition of STP. For the case $q = q^H$, $\tilde{V}_{t-1}^*(q^H, x) = 0$ is true as $\tau_t^Y(y'; q^H, x) = 0$ according to the definition of STP. ■

Based on the result above, we could perform backward recursion within a smaller state space. However, the summation $\sum_{y' \in Y} \tau^Q(q'|q, x', y') \tau_t^Y(y'; q, x)$ is computationally intractable. Our next contribution is to devise a tractable Monte Carlo method to compute the evolution of the environment state.

3.3 Approximating the uncertainty evolution with Monte Carlo approach

To simplify (3), we define the compact transition kernel

$$\tau_t^Q(q'|q, x', x) := \sum_{y' \in Y} \tau^Q(q'|q, x', y') \tau_t^Y(y'; q, x), \quad (4)$$

to indicate the probability of the simplified automaton state transiting to q' from q at x' using STP. While solving (3), we are only interested in $\tau_t^Q(q'|q, x', x)$ rather than the value of $\tau_t^Y(y'; q, x)$ for each y' . Thus, we rewrite (3) using the compact transition kernel as

$$\tilde{V}_{t-1}^*(q, x) = 1_G(q) + 1_{S \setminus G}(q) \max_{u \in U} \sum_{q' \in Q} \sum_{x' \in X} \tilde{V}_t^*(q', x') \tau^X(x'|x, u) \tau_t^Q(q'|q, x', x).$$

Now our objective is to find a computationally tractable way to obtain $\tau_t^Q(q'|q, x', x)$. First, we obtain the probability of a cell being inside the hazardous area using STP. We denote the probability of $x' \in \gamma(y_t)$ for all neighbor pairs (x, x') by the following quantity:

$$p_t(x'; \gamma, q) = \begin{cases} \sum_{y_t \in Y} 1_{\gamma(y_t)}(x') \tau_t^Y(y_t | [y_{t-1}]_x = 0), & q \neq q^H \\ 1, & q = q^H. \end{cases} \quad (5)$$

Then, the compact kernel τ_t^Q can be obtained as follows. First, when $q' = q^H$, the above quantity (5) is also the probability of $x' \in \gamma^{q^H}(y_t)$. Thus, $\tau_t^Q(q^H|q, x', x) = p_t(x'; \gamma, q)$. Second, for the state where $q' \neq q^H$, the probability of $x' \in \gamma^{q'}(y_t)$ is computed as the product of the probability of x' not being contaminated and the probability of the robot at a certain subset of X_f determined by q and q' . For example, in Figure 2, $\tau_t^Q(q^1|q^1, x', x) = (1 - p_t(x'; \gamma, q))(1 - \mathbf{1}_B(x'))$.

The quantity (5) is still costly to compute for $q \neq q^H$, but its approximation $\tilde{p}_t(x'; \gamma, q)$ can be obtained by a Monte Carlo method as shown in algorithm 1 below.

Algorithm 1 Monte Carlo approximation of $p_t(x'; \gamma, q)$

- 1: **Input:** robot state pair (x, x') , where $x' \in N(x)$; time t ; total episode number E , hazard evolution episodes $\{y_0, y_1^e, \dots, y_N^e\}$, where $0 \leq e \leq E$.
 - 2: **Initialize:** loop counter $n = 0$; contaminated counter at $t - 1$, $c_{t-1} = 0$; contaminated counter at t , $c_t = 0$
 - 3: **while** $n \leq E$ **do**
 - 4: obtain y_t^n and a y_{t-1}^n from episode $\{y_0, y_1^n, \dots, y_N^n\}$
 - 5: **if** $[y_{t-1}^n]_x = 0$ **then**
 - 6: $c_{t-1} = c_{t-1} + 1$
 - 7: **if** $[y_t^n]_{x'} = 1$ **then**
 - 8: $c_t = c_t + 1$
 - 9: **end if**
 - 10: **end if**
 - 11: $n = n + 1$
 - 12: **end while**
 - 13: **Output:** $\tilde{p}_t(x'; \gamma, q) := c_t / c_{t-1}$ for $q \neq q^H$.
-

Above, $\tilde{p}_t(x'; \gamma, q)$ is obtained for all neighbor pairs $(x, x') \in X_f \times X_f$ at all $t \leq N$. Then, we obtain the approximated (4) as $\tilde{\tau}_t^Q(q'|q, x', x)$, where the $2^{|X|}$ summation due to the size of Y is replaced with algorithm 1.

3.4 Solving the backward recursion tractably

By recursively solving (5) with $\tilde{\tau}_t^Q(q'|q, x', x)$, we approximately obtain the value functions $\tilde{V}_{t-1}^*(q, x)$ for all the states (q, x) and for $t = N, \dots, 0$.

To summarize the approaches introduced so far, we firstly simplified (1) to (3) by decoupling the hazard process $\tau_t^Y(y'; q, x)$ using STP. Then, we approximated the combined kernel $\tau_t^Q(q'|q, x', x)$ using $\tilde{p}_t(x'; \gamma, q)$ from Monte Carlo simulations to solve (3) in a tractable way as (5). Note that we lose accuracy in the proposed approach to alleviate the computation complexity during two processes. One is the decoupling of the environment state from the backward recursion. Although we use STP to capture the coupling relation as in Theorem 2, we still cannot reach the exact result as solving (1). The other one is the Monte Carlo approximation of the hazard process. This inaccuracy can be mitigated by using a large number of Monte Carlo samples in algorithm 1 as it is an offline computation.

4 Case study

We present case studies of planning in an uncertain environment to compare the performance of our method with the most capable existing approaches to address dynamical uncertainties as far as we are concerned. We show that our method provides trajectories with a higher probability of safety than D* Lite and the method in [38] for point-to-point and complex mission planning, respectively.

We use an evolving fire based on [31] as an example of the dynamical uncertainties in the following experiments. Starting from an initial location, the fire spreads to the neighboring grid cells probabilistically. Particularly, the transition kernel of the set process $\tau^Y : Y \times Y \rightarrow [0, 1]$ is

$$\tau^Y(y_{t+1}|y_t) = \prod_{x^1=0}^{m-1} \prod_{x^2=0}^{n-1} \Pr_f\{[y_{t+1}]_x | y_t\}, \quad (6)$$

$$\Pr_f\{[y_{t+1}]_x | y_t\} := \begin{cases} p_n(x, y_t) & \text{if } [y_{t+1}]_x = 0 \wedge [y_t]_x = 0 \\ 1 - p_n(x, y_t) & \text{if } [y_{t+1}]_x = 1 \wedge [y_t]_x = 0 \\ 0 & \text{if } [y_{t+1}]_x = 0 \wedge [y_t]_x = 1 \\ 1 & \text{if } [y_{t+1}]_x = 1 \wedge [y_t]_x = 1, \end{cases} \quad (7)$$

where $p_n(x, y_t)$ denotes the probability of a safe grid cell x remaining safe in the next time step, and is defined as:

$$p_n(x, y_t) = \left(1 - p_f(x)\right)^{N_f(x, y_t)} \left(1 - \frac{p_f(x)}{\sqrt{2}}\right)^{D_f(x, y_t)}. \quad (8)$$

For each grid x , $p_f(x)$ is a constant describing the evolving speed of the hazard in different types of the space. Parameters N_f and D_f in (8) are the numbers of direct and diagonal neighbors inside the hazardous areas. For diagonal neighbors, we scale the parameter $p_f(x)$ by $\frac{1}{\sqrt{2}}$, which indicates the flexibility to set different hazard evolving speed based on the distance of the neighboring cell. The hazard model described by (6), (7) and (8) can be intuitively explained as follows. From (7), a contaminated cell remains contaminated in all following time steps. The probability of an uncontaminated cell getting contaminated, i.e., affected by fire, in the next time step is independently affected by the environment states of its neighbors at the current time step, as shown in (8). Using this process, we obtain $[y_{t+1}]_x$, for $\forall x$. Then, the probability of any environment state y_{t+1} can be computed by multiplying the probability of each grid cell x holding the value of $[y_{t+1}]_x$ given y_t as in (6).

4.1 Comparison to D* Lite in point-to-point planning

We set up a point-to-point planning case by simplifying the complex mission we use in Figure 1. We only kept one target A and three initial hazards H_1 , H_2 and H_3 . The objective is to plan a path from the initial location I to target A while maximizing the probability of finishing this mission safely. The setup is shown in Figure 3(a). As discussed in the introduction, D* Lite is a reactive replanning algorithm to find the shortest path in dynamical environments. The initially planned path is adjusted when an obstacle is detected within the robot’s visibility during the execution of the path. We set the visibility of the robot to be its neighbor grids within 2 steps. For our method, we used the offline closed-loop policy obtained from (5) and did not adjust it during execution. Note we obtain optimal control inputs for any given state at any time accounting for the dynamical uncertainties, while D* Lite only considers a static environment. In Figure 3(a), the initial D* Lite path (the green line) directly went to the target as it does not consider the dynamical model of the environment, while the path planned by our method (the blue line) took a detour. We tested the two methods by simulating the hazard spreading process 1000 times to compare their empirical probabilities of safely reaching the target, as shown in the table.

Method	D* Lite	This work
Success rate	30.0%	38.7%

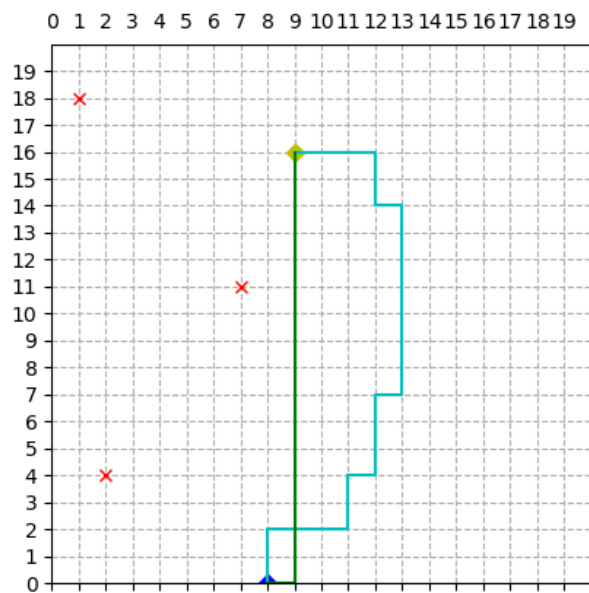
For D* Lite, the robot sometimes backtracked its path when blocked by hazard during the execution. One example is shown in Figure 3(b) and Figure 3(c), where the path already traveled is solid, and the path planned but not yet executed is dotted. At $t = 18$, the robot was at $(12, 10)$ and planned to go to $(13, 10)$ at the next time step as in Figure 3(b). But at $t = 19$, the robot observed that $(13, 10)$ was blocked by hazard and it had to go back to $(12, 9)$ and plan a new path as in Figure 3(c). At $t = 19$, the robot had already reached $(13, 14)$ using our method as in Figure 3(d). Note that this might happen several times, like in the example, and the robot using D* Lite will have a much longer detour. Then, the hazard could contaminate more spaces and decrease the probability of finishing the task safely.

4.2 Comparison to past works in complex mission planning

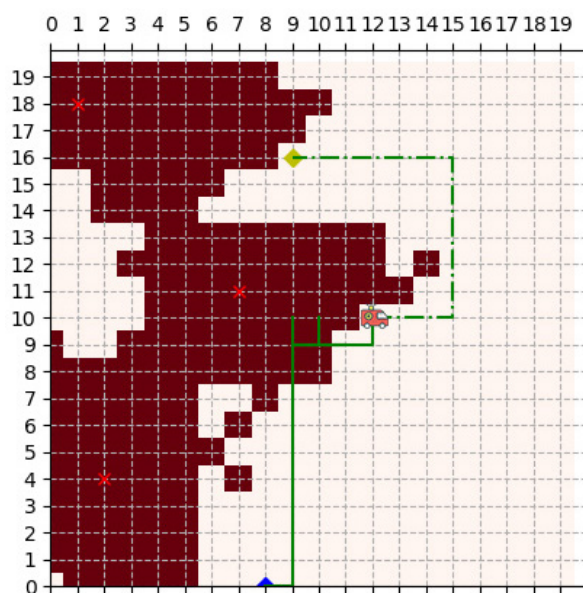
We compare our method to [38], where a similar DP is developed but the coupling between state elements is ignored. We show that we obtain a better path using our method in terms of the probability of satisfying the specification safely than the method in [38]. In this example, we used the specification of reaching the target A first and exiting at F , while avoiding the hazard H . Some obstacles were added to increase the complexity of the task. The time consumption to solve the DP for our method and the method in [38] were both approximated 150 seconds. The paths obtained offline using both methods are shown in Figure 4, where our method chose a longer but safer path. Note that red color indicates that the grid cell was contaminated at a certain time, where darker red means the cell was contaminated earlier. We show the empirical success rates for both methods in the following table.

Method	DP without STP in [38]	This work
Success rate	0.003%	0.667%

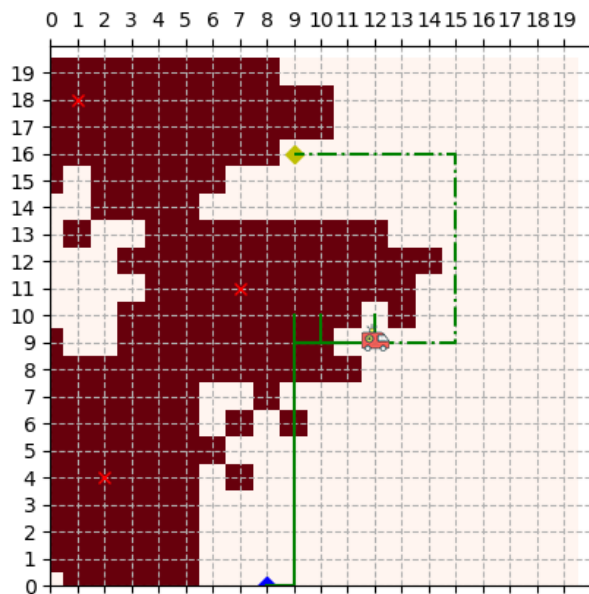
Note that for the case studies, due to the size of the environment state space, we could not obtain a ground truth for the planning problem, and thus had to use the approximated approaches. The algorithms were coded in Python 3.6 on a computer with 2.3 GHz i5 quadcore processor and 16 GB memory.



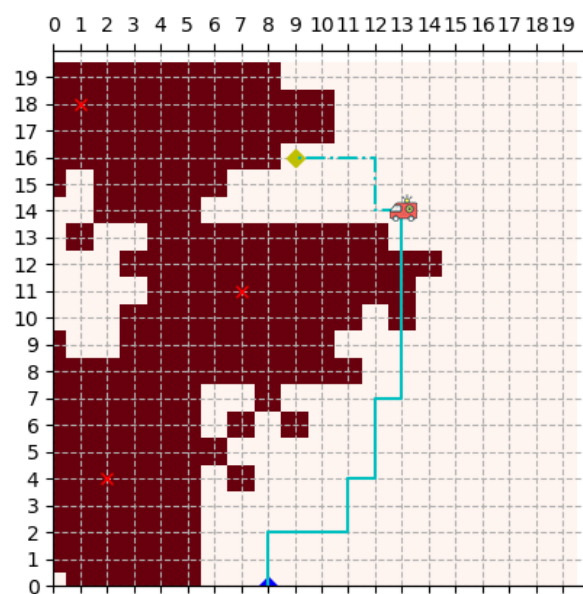
(a) Initial path by both methods



(b) D* Lite at $t = 18$



(c) D* Lite at $t = 19$



(d) Our method at $t = 19$

Figure 3: The paths from D* Lite and our method are shown in green lines and blue lines, respectively. Path already traveled is solid and not yet executed is dotted. Because D* Lite backtracked and replanned several times as in (b) and (c), our method yielded better path than D* Lite under dynamical uncertainties as in (d).

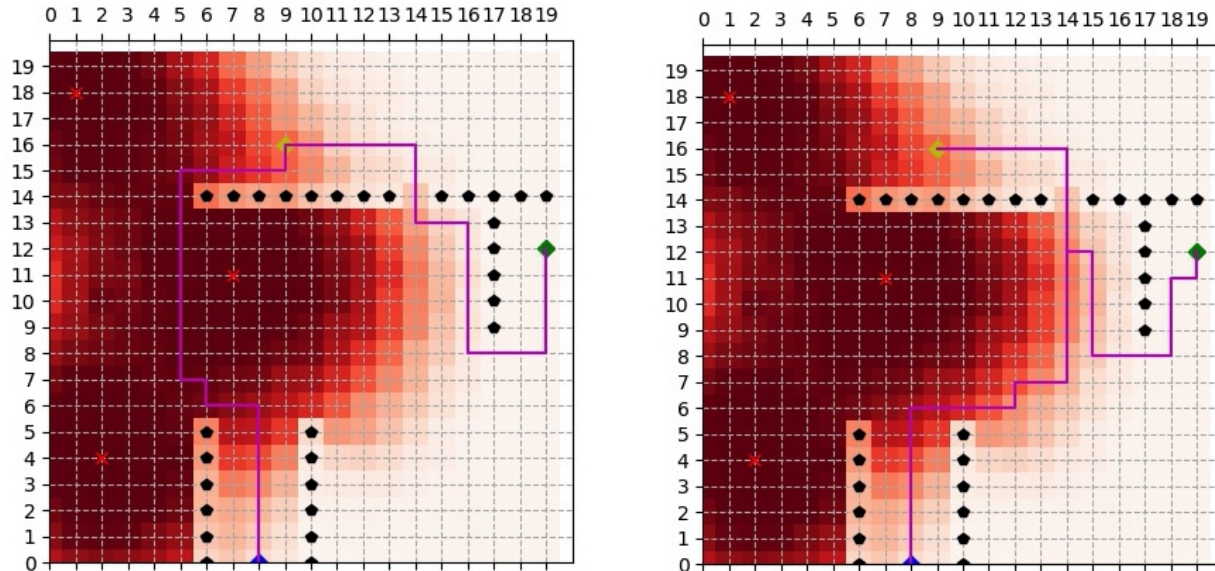


Figure 4: Optimal path from [38] (left) and our method (right). In this case our method planned a longer but safer path, which outperformed the method in [38].

5 Conclusions

We developed a framework of safe planning for complex missions in uncertain dynamical environments. To address computational tractability, we devised a method to decouple the computation of the environment’s dynamical evolution from that of the robot trajectory to solve the planning problem, based on a novel notion of safe transition probability. Also, we mitigated the intractability issue by designing a Monte Carlo approximation for the environment’s uncertainty propagation. We showed in case studies that our method outperforms existing approaches for planning under dynamical uncertainties.

Future directions include combining the idea of replanning approach employed in such as D* Lite in our framework. This enables incorporating online observations and making corresponding online adjustments to improve the safety probability further. Furthermore, we plan to run the methodology on realistic robotics testbeds.

Acknowledgement

The authors thank Dr. David Adjashvili for helpful discussions.

References

- [1] Georges S Aoude, Brandon D Luders, Joshua M Joseph, Nicholas Roy, and Jonathan P How. Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns. *Autonomous Robots*, 35(1):51–76, 2013.
- [2] Andrea Bajcsy, Somil Bansal, Eli Bronstein, Varun Tolani, and Claire J Tomlin. An efficient reachability-based framework for provably safe autonomous navigation in unknown environments. *arXiv preprint arXiv:1905.00532*, 2019.

- [3] Joseph L Baxter, EK Burke, Jonathan M Garibaldi, and Mark Norman. Multi-robot search and rescue: A potential field based approach. In *Autonomous robots and agents*, pages 9–16. Springer, 2007.
- [4] Zoltán Beck, Luke Teacy, Alex Rogers, and Nicholas R Jennings. Online planning for collaborative search and rescue by heterogeneous robot teams. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 1024–1033. International Foundation for Autonomous Agents and Multiagent Systems, 2016.
- [5] Sebastian Brechtel, Tobias Gindele, and Rüdiger Dillmann. Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 392–399. IEEE, 2014.
- [6] Jennifer Casper and Robin R. Murphy. Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 33(3):367–385, 2003.
- [7] Yushan Chen, Jana Tumová, and Calin Belta. LTL robot motion control based on automata learning of environmental dynamics. In *2012 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5177–5182. IEEE, 2012.
- [8] Hao Cheng and George V Hadjisophocleous. Dynamic modeling of fire spread in building. *Fire Safety Journal*, 46(4):211–224, 2011.
- [9] Igor Cizelj, Xu Chu Dennis Ding, Morteza Lahijanian, Alessandro Pinto, and Calin Belta. Probabilistically safe vehicle control in a hostile environment. *IFAC Proceedings Volumes*, 44(1):11803–11808, 2011.
- [10] Edmund M Clarke Jr, Orna Grumberg, Daniel Kroening, Doron Peled, and Helmut Veith. *Model checking*. MIT press, 2018.
- [11] Donato Di Paola, Annalisa Milella, Grazia Cicirelli, and Arcangelo Distante. An autonomous mobile robotic system for surveillance of indoor environments. *International Journal of Advanced Robotic Systems*, 7(1):8, 2010.
- [12] Dave Ferguson, Anthony Stentz, and Sebastian Thrun. PAO for planning with hidden state. In *2004 IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2840–2847. IEEE, 2004.
- [13] Oren Gal, Zvi Shiller, and Elon Rimon. Efficient and safe on-line motion planning in dynamic environments. In *2009 IEEE International Conference on Robotics and Automation (ICRA)*, pages 88–93. IEEE, 2009.
- [14] Gokce Gorbil, Avgoustinos Filippoupolitis, and Erol Gelenbe. Intelligent navigation systems for building evacuation. In *Computer and Information Sciences II*, pages 339–345. Springer, 2011.
- [15] Meng Guo, Karl H Johansson, and Dimos V Dimarogonas. Revising motion planning under linear temporal logic specifications in partially known workspaces. In *2013 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5025–5032. IEEE, 2013.

- [16] Sylvia L Herbert, Mo Chen, SooJean Han, Somil Bansal, Jaime F Fisac, and Claire J Tomlin. Fastrack: a modular framework for fast and guaranteed safe motion planning. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 1517–1522. IEEE, 2017.
- [17] Maryam Kamgarpour, Tony A Wood, Sean Summers, and John Lygeros. Control synthesis for stochastic systems given automata specifications defined by stochastic sets. *Automatica*, 76:177–182, 2017.
- [18] Shinji Kawatsuma, Mineo Fukushima, and Takashi Okada. Emergency response by robots to fukushima-daiichi accident: summary and lessons learned. *Industrial Robot: An International Journal*, 39(5):428–435, 2012.
- [19] Sven Koenig and Maxim Likhachev. D* Lite. *AAAI/IAAI*, 15, 2002.
- [20] Hadas Kress-Gazit, Tichakorn Wongpiromsarn, and Ufuk Topcu. Correct, reactive, high-level robot control. *IEEE Robotics & Automation Magazine*, 18(3):65–74, 2011.
- [21] Vijay Kumar, Daniela Rus, and Sanjiv Singh. Robot and sensor networks for first responders. *IEEE Pervasive computing*, 3(4):24–33, 2004.
- [22] Morteza Lahijanian, Sean B Andersson, and Calin Belta. Formal verification and synthesis for discrete-time stochastic systems. *IEEE Transactions on Automatic Control*, 60(8):2031–2045, 2015.
- [23] Maxim Likhachev and Anthony Stentz. Probabilistic planning with clear preferences on missing information. *Artificial Intelligence*, 173(5-6):696–721, 2009.
- [24] Brandon Luders, Mangal Kothari, and Jonathan How. Chance constrained RRT for probabilistic robustness to environmental uncertainty. In *AIAA Guidance, Navigation, and Control Conference*, page 8160, 2010.
- [25] Matthew R Maly, Morteza Lahijanian, Lydia E Kavraki, Hadas Kress-Gazit, and Moshe Y Vardi. Iterative temporal motion planning for hybrid systems in partially unknown environments. In *Proceedings of the 16th International Conference on Hybrid Systems: computation and control*, pages 353–362. ACM, 2013.
- [26] Michael Otte and Emilio Frazzoli. RRT^X: Asymptotically optimal single-query sampling-based motion planning with quick replanning. *The International Journal of Robotics Research*, 35(7):797–822, 2016.
- [27] Mike Phillips and Maxim Likhachev. SIPP: Safe interval path planning for dynamic environments. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5628–5635. IEEE, 2011.
- [28] Shahar Sarid, Bingxin Xu, and Hadas Kress-Gazit. Guaranteeing high-level behaviors while exploring partially known maps. In *Robotics*, pages 377–384. MIT Press, 2013.
- [29] Andreas Schadschneider, Wolfram Klingsch, Hubert Klüpfel, Tobias Kretz, Christian Rogsch, and Armin Seyfried. Evacuation dynamics: Empirical results, modeling and applications. *Encyclopedia of Complexity and Systems Science*, pages 3142–3176, 2009.
- [30] Dylan A Shell and Maja J Matarić. Insights toward robot-assisted evacuation. *Advanced Robotics*, 19(8):797–818, 2005.

- [31] H elene Soubaras. Risk prediction in a space-time markov model applied to propagating contamination. In *Proceedings of IPMU*, volume 8, page 561, 2008.
- [32] Anthony Stentz. Optimal and efficient path planning for partially known environments. In *Intelligent Unmanned Ground Vehicles*, pages 203–220. Springer, 1997.
- [33] Sean Summers, Maryam Kamgarpour, John Lygeros, and Claire Tomlin. A stochastic reach-avoid problem with random obstacles. In *Proceedings of the 14th International Conference on Hybrid Systems: computation and control*, pages 251–260. ACM, 2011.
- [34] Sean Summers and John Lygeros. Verification of discrete time stochastic hybrid systems: A stochastic reach-avoid decision problem. *Automatica*, 46(12):1951–1961, 2010.
- [35] Alphan Ulusoy, Tichakorn Wongpiromsarn, and Calin Belta. Incremental controller synthesis in probabilistic environments with temporal logic constraints. *The International Journal of Robotics Research*, 33(8):1130–1144, 2014.
- [36] Eric M Wolff, Ufuk Topcu, and Richard M Murray. Robust control of uncertain Markov decision processes with temporal logic specifications. In *2012 IEEE 51st Conference on Decision and Control (CDC)*, pages 3372–3379. IEEE, 2012.
- [37] Tichakorn Wongpiromsarn, Ufuk Topcu, and Richard M Murray. Receding horizon control for temporal logic specifications. In *Proceedings of the 13th ACM International Conference on Hybrid Systems: computation and control*, pages 101–110. ACM, 2010.
- [38] Tony A Wood and Maryam Kamgarpour. Automaton-based stochastic control for navigation of emergency rescuers in buildings. In *2016 IEEE Conference on Control Applications (CCA)*, pages 587–592. IEEE, 2016.
- [39] Bingyu Zhou, Wilko Schwarting, Daniela Rus, and Javier Alonso-Mora. Joint multi-policy behavior estimation and receding-horizon trajectory planning for automated urban driving. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2388–2394. IEEE, 2018.