Rapid Multirobot Deployment with Time Constraints

Stefano Carpin Marco Pavone Brian M. Sadler

Abstract— **In this paper we consider the problem of multirobot deployment under temporal deadlines. The objective is to compute strategies trading off safety for speed in order to maximize the probability of reaching a given set of target locations within a given temporal deadline. We formulate this problem using the theory of Constrained Markov Decision Processes and we show that thanks to this framework it is possible to determine deploying strategies maximizing the probability of success while satisfying a temporal deadline. Moreover, the formulation allows to *exactly* compute the failure probability of complex deployment tasks. Simulation results illustrate how the proposed method works in different scenarios and show how informed decisions can be made regarding the size of the robot team.**

I. Introduction

In this paper we consider the problem of multirobot deployment under temporal deadlines. Our focus is one large teams (swarms) of small, inexpensive robots with limited capabilities [5], [21]. In this case robots can be considered expendable assets and one can opt for more risky control strategies accepting the possibility that some robots may break down during the task. In fact, there has been increased interest in developing techniques for aggressive robot maneuvering [13], [18]. From this standpoint, a tradeoff between speed and risk is evident, because when robots move very fast in their environment the possibility of collisions and irremediable damage increases. In these situations it is therefore necessary to strike a balance between velocity and risk. The deployment problem we consider is defined as follows. A set of K robots is placed at a given location and a set of N target locations is provided. The team objective is to deploy so that eventually each location is reached by at least one robot. Once a robot reaches a target location it stops. A temporal constraint is given, i.e., each robot shall terminate its operation within the given deadline. Problems like this find application in situations where robots are sent into a region to acquire time-sensitive information before human operators enter the scene. Examples include urban search and rescue, surveillance, and the like. In a situation like this each robot faces a time/safety tradeoff. Robots have an incentive to move as fast as possible to reach one of the locations before the given deadline. At the same time, chances of success decrease when moving at high speed because a robot moving at high velocity is more likely to incur in an event precluding its success (e.g., the robot bumps into an obstacle and breaks down, or fails to properly localize itself and gets lost, etc.) In the deployment situation, the redundancy in the robot team is essential to mitigate this problem. In other words, by increasing the number K of robots one can operate at higher velocities and tolerate that some robots will fail to navigate to their assigned location, as long as the team as a whole succeeds. Relevant problems in this scenario are then 1) deciding how many robots should be in the team to guarantee success with a certain level of confidence given a certain temporal deadline; 2) assigning target locations to robots; 3) for each robot computing a strategy to reach the assigned location before the deadline and maximizing the chances of success. Our eventual long term objective is to characterize the space relating deployment policies, temporal deadlines, success probability, the complexity of the environment, and the size of the team. In our recent work [7] we have defined this problem and we have proposed two very simple algorithms whose performance can be characterized only in approximate terms. A first step towards validating linking our theoretical model with real world hardware is presented in [22]. In this paper we extend our preliminary findings by using the theory of constrained Markov decision processes (CMDPs). By using formulating our problem using CMDPs, we obtain the two new results:

- we compute *provably optimal* strategies that satisfy in expectation a temporal deadline;
- we compute the *exact* failure probability for the optimal strategy.

Our theoretical findings are corroborated by various simulations. Results derived from this formulation is operationally relevant because they can inform decision makers about the relationship between team size, probability of success and the temporal deadline.

Although we embrace a different solving approach, we retain two of the hypotheses we formerly made. The first is that we aim at minimalistic algorithms that can be run on large teams of simple robots. These robots are supposed to predominantly operate on their own, without exchanging large amount of information with other team members or knowing how large the team is. The rationale behind this choice is that large teams of simple robots are easier to deploy (no complex setup), more resilient and more cost effective. In this paper we assume that robots do not exchange any information before or during the mission. The second assumption is that the robots know the map of the environment and can self localize themselves. Both elements can be achievable in certain conditions. For example, indoor maps are becoming ubiquitous (e.g., Google indoor maps)

S. Carpin is with the School of Engineering, University of California-Merced, CA, USA.

M. Pavone is with the Department of Aeronautics and Astronautics, Stanford University, CA, USA.

B.M. Sadler is with the Army Research Lab, Adelphi, MD, USA.

and outdoor terrain maps can be obtained through aerial recognition. Localization in outdoor domains is also achievable (e.g., GPS), and our recent paper [22] discusses the impact of localization errors in indoor environments. Therefore in this manuscript we keep the simplifying assumption that robot location is known and we focus on the problem of balancing risk and performance. Finally, we stress that we are targeting small platforms, whose individual performances can be assumed to be mutually independent. This way, even if one or more robots fail on their way to a target, they do not obstruct the way to other robots.

The rest of the paper is organized as follows. Section II offers brief pointers to related literature, mostly outlining differences between our problem and previous work. The theory of constrained Markov decision processes is introduced in section III, and in Section IV we show how to formulate a deployment problem using the theory of constrained Markov decision processes. Simulations are presented and discussed in Section V, while conclusions and future work are offered in Section VI.

II. Related work

The deployment task we consider is related to various multirobot problems in the area of surveillance and information gathering. However, to the best of our knowledge the specific problem we study is new.

Deployment is related to coverage control [9], [24], an area that has received enormous attention in recent years (see [6] for a comprehensive literature review.) These methods are mostly based on local optimization and aim at control strategies steering the system towards an asymptotic equilibrium point optimizing a given performance function. On the contrary, in this paper we focus on the constraints on the transient stage of the solution. Approaches based on random dispersion were proposed too [19], [20], but these methods do not offer performance bounds, neither for probability of success nor deployment time. Temporal constraints were considered in [7], [10], [15], though in a different setting. Approaches relying communication, often modeled using POMDPs have also been proposed [3], [12], [17], but in this paper we assume agents do not exchange any information. Markov decision processes have been extensively used in robotics and any list is necessarily incomplete. However, the use of Constrained Markov Decision Processes [2] in robotics is much more limited, in particular when it comes to deployment-like problems [11].

To the best of our knowledge, the deployment problem under temporal constraints we study in this paper was first introduced in our former papers [7], [22] and the use of Constrained Markov Decision Processes to compute deployment strategies under temporal constraints is new.

III. Total Cost Constrained Markov Decision Processes

In this section we introduce the notation used in the remaining of this paper. We provide a brief summary about total cost Markov Decision processes and we then outline key differences with Constrained Markov Decision Processes. The reader is referred to [4] for a thorough discussion.

A. Total Cost Markov Decision Processes

A discrete Markov Decision Process (MDP) is a model used to study decision making problems where actions outcomes are stochastic and the state is observable. A stationary MDP is defined by a quadruple $\mathbf{X}, A, c, \mathcal{P}$ where:

- $\mathbf{X}$ is a finite set of $n = |\mathbf{X}|$ states. The evolution over time of the state of an MDP is stochastic and the state at time t is given by the random variable X_t.
- A is a collection of n finite sets. $A(x)$ is the set of actions that can be applied when the MDP is in state x. It is convenient to define the set $\mathcal{K} = \{(x, a) : x \in \mathbf{X}, a \in A(x)\}$. In general, the action taken at time t is a random variable indicated by the letter A_t.
- $c : \mathcal{K} \to \mathbb{R}$ is the function defining the *immediate* cost incurred when applying action $a \in A(x)$ while in state $x \in \mathbf{X}$.
- $\mathcal{P}^a_{xy}$ is the one step transition probability from state x to state y when action a is applied, i.e., $\mathcal{P}^a_{xy} = \Pr[X_{t+1} = y | X_t = x, A_t = a]$.

Based on the above definitions, the sequence of states and actions over time are defined by a stochastic process that we will indicate as (X_t, A_t). The above model is *stationary* because costs and transition probabilities do not depend on time. Solving an MDP means determining a policy π defining which action should be applied at time t given that $X_t = x$ in order to minimize a given cost function. In general $\pi(x)$ needs not to be a function into $A(x)$ but could in general be a mass distribution over $A(x)$. Different cost functions have been defined in MDP related literature: finite horizon; infinite horizon with discounted cost; average-cost with infinite horizon; and total cost. For our application the most appropriate model is the *total cost*. For this model to be valid, one needs to make the following further assumptions:

1) $\mathbf{X}$ is partitioned into sets $\mathbf{X}'$ and $\mathbf{M}$, i.e., $\mathbf{X} = \mathbf{X}' \cup \mathbf{M}$ and $\mathbf{X}' \cap \mathbf{M} = \emptyset$.
2) For each $x \in \mathbf{M}$, $c(x, a) = 0$
3) The MDP is *transient*, i.e., $y \in \mathbf{M}, x \in \mathbf{X}' \Rightarrow \mathcal{P}^a_{xy} = 0$, and for every policy π and every state $x \in \mathbf{X}'$, $\sum_{t=1}^{\infty} \Pr^{\pi}[X_t = x] < \infty$ where $\Pr^{\pi}[X_t = x]$ is the probability that the MDP is in state x at time t while following policy π.

The third requirement imposes a special structure on the underlying MDP that ensures that the following cost exists and is finite:

$$c(\pi) = E_{\pi}\left[\sum_{t=1}^{+\infty} c(X_t, A_t)\right]$$

where E_{π} indicates the expectation with respect to the probability mass over the sequence (X_t, A_t) induced by π. Indeed, the transient property ensures that eventually the state will enter and remain in $\mathbf{M}$ where no more cost is accrued. Therefore $c(\pi)$ is finite even though it is an infinite sum of undiscounted costs. Let Π be the set of all policies for a given MDP. We define

$$\pi^* = \arg\min_{\pi\in\Pi} c(\pi).$$

It is well known that given a stationary MDP there exists an optimal, stationary, deterministic Markovian policy π^*. Therefore the optimal policy is a function $\pi^* : \mathbf{X} \to A$. Optimal policies can be computed in different ways, and dynamic programming methods (value iteration or policy iteration) are the techniques most commonly used.

B. Total Cost Constrained Markov Decision Processes

A total cost Constrained Markov Decision Process (CMDP) extends the MDP model by introducing additional costs and associated constraints. A CMDP is defined by $\mathbf{X}, A, c, \mathcal{P}, d_i, D$ where $\mathbf{X}, A, c, \mathcal{P}$ are defined as above and define a transient MDP. Furthermore:

- $d_i : \mathcal{K} \to \mathbb{R}$, with $1 \le i \le L$ is a family of L *additional* costs incurred when applying action a from state x. In order to set the stage for a sound definition of total costs, we furthermore assume that for each $x \in \mathbf{M}$, $d(x, a) = 0$.
- $D \in \mathbb{R}^L$ is a vector of L upper bound values for the expected cumulative d_i costs.

Before defining the total cost for CMDP it is necessary to recall that for CMDPs the optimal policy in general depends on the probabilistic distribution of the initial state. In the following this distribution is indicated with the letter β, with the understanding that $\beta(x) = \Pr[X_1 = x]$ for $x \in \mathbf{X}$. For a CMDP we define the total costs:

$$c(\pi, \beta) = E_{\pi,\beta}\left[\sum_{t=1}^{+\infty} c(X_t, A_t)\right]$$

$$d_i(\pi, \beta) = E_{\pi,\beta}\left[\sum_{t=1}^{+\infty} d_i(X_t, A_t)\right] \quad 1 \le i \le L.$$

Solving a CMDP means determining an optimal policy π^* minimizing the expected total cost $c(\pi, \beta)$ while ensuring that each of the additional L total costs defined by the functions d_i is (in expectation) bounded by D_i, i.e.,

$$\begin{aligned} \pi^* &= \min_{\pi} c(\pi, \beta) \\ \text{s.t. } & d_i(\pi, \beta) \le D_i, \quad 1 \le i \le L. \end{aligned} \tag{1}$$

CMDPs do not share many of the properties enjoyed by MDPs (see [2] for a comprehensive discussion of the subject.) For example, even for the simplest stationary case the optimal policy for a CMDP may require randomization and be non deterministic. In addition, CMDPs cannot be solved using dynamic programming but are rather solved using linear programming[1].

[1]To be precise there exist more than one LP formulation that can be used, and methods based on Lagrange multipliers have been introduced too. However, they will not be considered in this paper.

A fundamental theorem concerning CMDPs [1] relates the solution of the optimization problem defined in Eq. 1 to the constrained linear optimization problem defined as follows. First, let $\mathcal{K}' = \{(x, a), x \in \mathbf{X}', a \in A(x)\}$. Next, let us introduce $|\mathcal{K}'|$ optimization variables $\rho(x, a)$, each one associated with an element in $\mathcal{K}'$. Let $\delta_x(y) = 1$ when $x = y$ and 0 otherwise. Then, consider the following constrained linear optimization problem:

$$\begin{aligned} \min & \sum_{(x,a)\in\mathcal{K}'} \rho(x, a) c(x, a) \\ \text{s.t.} & \sum_{(x,a)\in\mathcal{K}'} \rho(x, a) d_i(x, a) \le D_i \quad 1 \le i \le L \\ & \sum_{y\in\mathbf{X}'} \sum_{a\in A(y)} \rho(y, a)(\delta_x(y) - \mathcal{P}^a_{xy}) = \beta(x) \ \forall x \in \mathbf{X}' \\ & \rho(x, a) \ge 0. \end{aligned} \tag{2}$$

The constrained optimization problem defined in Eq. 1 has a solution if and only if the problem defined in Eq. 2 is feasible [1], and the optimal solution to the linear program induces an optimal stationary, randomized policy for the CMDP defined as follows:

$$\pi^*(x, a) = \frac{\rho(x, a)}{\sum_{a\in A(x)} \rho(x, a)} \quad x \in \mathbf{X}', a \in A(x) \tag{3}$$

where $\pi^*(x, a)$ is the probability of taking action a when in state x. If the denominator of Eq. 3 is 0, then the policy can be arbitrarily defined for (x, a). Note that the policy is not defined for states in $\mathbf{M}$ because it is assumed that the evolution terminates when the state enters $\mathbf{M}$.

The optimization variables $\rho(x, a)$ can be interpreted as *occupancy measures*, where the occupancy measure of a couple (x, a) is defined as

$$\rho(x, a) = \sum_t P[X_t = x, A_t = a] \tag{4}$$

where the probability is implicitly conditioned on a policy π and an initial distribution β. Note that the occupancy measure is a sum of probabilities, but in general is not a probability itself. However, in the next section we prove that, because of the way we define our model, one of the $\rho(x, a)$ variables provides the exact probability that a robot fails to accomplish its task.

IV. Deployment using CMDs

In this section we formalize the deployment problem using a graph model and we show how given a deployment instance it is possible to build an associated CMDP. As we did in [7], we model the environment using an undirected graph $G = (X, E)$ where X is the set of vertices and E is the set of edges. Graph abstractions are common for this kind of problems, and we have in the past shown how it is possible to automatically extract graphs from occupancy grid maps and associate to the graph elements quantities related to the underlying metric maps, like traversal costs and so on [16].

One vertex $d \in X$ represents the deployment site where robots start from, whereas the set of target locations is $T \subset X$. An edge $e \in E$ between two vertices v_1 and v_2 means it is possible to go form v_1 to v_2 and viceversa. To each edge $e_i \in E$ we associate a function $S_i : \mathbb{R}^+ \rightarrow [0,1]$. $S_i(t)$ is the probability of successfully completing a move along edge e_i as a function of the time spent during the move. S_i then captures the tradeoff between speed and risk and is subject to the boundary constraints $S_i(0) = 0$ and $\lim_{t \to +\infty} S_i(t) = 1$. Moreover, S_i is a non decreasing function.

We associate an instance of the deployment problem to a CMDP as follows. The set of states is $\mathbf{X} = X \cup \{\mathcal{S}\}$ where $\mathcal{S}$ is a sink state introduced to model failures. Robots enter the sink state when they fail to perform a transition between two vertices and cease to operate. We partition the state set $\mathbf{X}$ into $\mathbf{M} = T$ and $\mathbf{X}' = \mathbf{X} \setminus \mathbf{M}$. With this partition, and using the transition probabilities defined in the following, the CMDP will be transient in $\mathbf{X}'$. The initial distribution β is defined as $\beta(d) = 1$ and $\beta(x) = 0$ for $x \neq d$. For each state $x \neq \mathcal{S}$ and $x \notin T$ we create $A(x)$ as follows. For each edge (x, y) we consider a discrete set of times based on the distance $l(x, y)$ between x and y. The discretization step Δ is fixed and equal for all edges. $A(x)$ is then defined as follows[2]:

$$A(x) = \{(y,t) : (x,y) \in E, 1 \leq t \leq \lceil l(x,y)/\Delta \rceil, t \in \mathbb{N}\}.$$

An action $(y,t) \in A(x)$ means that the robot tries to navigate from x to y spending at most time t. For the state $\mathcal{S}$ we define just one action $a_{\mathcal{S}} = (x_T, 0)$ where x_T is an arbitrary state in the target set T. We do not need to define actions for states in T because, as evidenced in the formulation of the optimization problem given in Eq. 2, these states and actions do not influence the solution. Finally, for each $x, y \in X$, $a \in A(x)$, we need to define the transition probabilities $\mathcal{P}^a_{xy}$. These probabilities are defined as follows:

$$\mathcal{P}^a_{xy} = \begin{cases} S_{(x,y)}(t) & \text{if } x, y \neq \mathcal{S}, a = (y,t) \in A(x) \\ 1 - S_{(x,z)}(t) & \text{if } x \neq \mathcal{S}, y = \mathcal{S}, a = (z,t) \in A(x) \\ 1 & \text{if } x = \mathcal{S}, y = x_T \in T, a = a_{\mathcal{S}} \\ 0 & \text{otherwise} \end{cases}$$

The first case of this equation models the probability of successfully completing in time t a move from x to y based on the S function associated with the edge. The second case instead models the failure probability, i.e., when a robot does not successfully make the transition from x to z it enters the sink state $\mathcal{S}$. The third case implies that once the system enters the sink state it deterministically[3] moves to one of the target states with probability 1. This deterministic transition into T makes the CMDP transient in $\mathbf{X}'$. Figure 1 shows the role of the sink state $\mathcal{S}$.

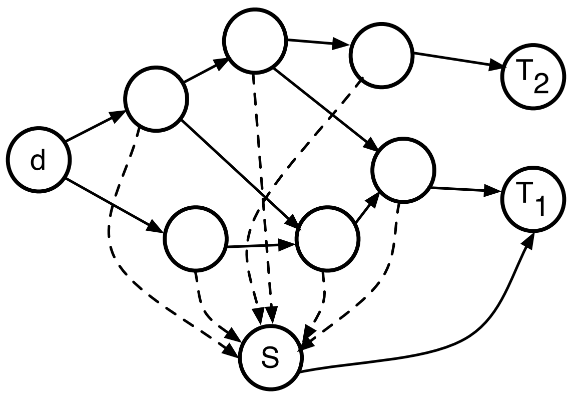


Fig. 1: Given a graph $G = (X, E)$ and a policy π, multiple stochastic paths from the deployment vertex d to the target vertices (T_1 and T_2 in the figure) may emerge. Each of them is associated with a failure probability defined as the probability that one of the transitions does not succeed. Whenever this happens the MDP enters $\mathcal{S}$ (dashed arrows) and then it deterministically moves into one state in T. As it will be illustrated in the following, the CMDP formulation we propose allows to exactly compute the probability that any path induced by the optimal policy π^* passes through $\mathcal{S}$.

To complete the definition of the CMDP we need to define the immediate costs $c(x, a)$, the additional costs $d_i(x, a)$ and the vector D. For all states x different from $\mathcal{S}$ we set $c(x, a) = 0$ for every action. For the sink state we set $c(\mathcal{S}, a_{\mathcal{S}}) = 1$. In this paper we consider just one additional cost, namely the time needed to complete an action. The cost is then $d(x, a) = t$ for $x \in \mathbf{X}'$, $a = (y, t) \in A(x)$, and 0 otherwise. With just one additional cost, we then need to provide just one bound so the vector D reduces to a single constant.

The following theorem establishes that for the special CMDP we just formulated the failure probability can be exactly computed.

Theorem 1: Consider a CMDP with the structure described in this section, let $\rho(x, a)$ the solution of the associated optimization problem given in Eq. 2, and let π^* the optimal policy derived from $\rho(x, a)$ using Eq. 3. Then, the probability of failure is equal to $\rho(\mathcal{S}, a_{\mathcal{S}})$ and

$$\rho(\mathcal{S}, a_{\mathcal{S}}) = c(\pi^*, \beta).$$

Proof. By definition (see Eq. 4),

$$\rho(\mathcal{S}, a_{\mathcal{S}}) = \sum_{t=1}^{+\infty} P[X_t = \mathcal{S}, A_t = a_{\mathcal{S}}]$$

where P is the probability induced by the policy and the initial distribution. Let Ω be sample space for the state evolution, and consider the family of events $B_t = \{X_t = \mathcal{S}\}$

[2]One could in fact put also a lower bound on t in the definition of $A(x)$. This makes sense from a practical point of view, but does not change the properties of the model, because actions with a too small t value will be associated with a 0 transition probability.

[3]Since $a_{\mathcal{S}} = (x_T, 0)$ is the only action defined for $\mathcal{S}$, it follows that once the state enters in $\mathcal{S}$ it will immediately move to x_T spending 0 additional time.

with $t = 1, 2, \ldots$. Clearly, for how we defined the transition probability, if one of the B_t is true then the deployment fails, and if the deployment fails one and only one of the B_t is true, because once the state enters $\mathcal{S}$ it deterministically moves to T at the next step and it stays there. Let us furthermore define the event $B = \Omega \setminus \bigcup_t B_t$. By definition, B and $\bigcup_t B_t$ are mutually exclusive, and also B_i and B_j are mutually exclusive for every $i \neq j$. Using the total probability theorem, the probability of the event failure ($\mathcal{F}$ in the following) is then:

$$\Pr[\mathcal{F}] = \sum_{t=1}^{+\infty} \Pr[\mathcal{F}|B_t] \Pr[B_t] + \Pr[\mathcal{F}|B] \Pr[B]$$

Because of our definitions, $\Pr[\mathcal{F}|B_t] = 1$ and $\Pr[\mathcal{F}|B] = 0$, so the expression simplifies to

$$\Pr[\mathcal{F}] = \sum_{t=1}^{+\infty} \Pr[B_t] = \sum_{t=1}^{+\infty} \Pr[X_t = \mathcal{S}, A_t = a_{\mathcal{S}}]$$

where we used the definition of B_t and the fact that only action $a_{\mathcal{S}}$ can be taken in $\mathcal{S}$. This proves the first part of the theorem. To prove the second statement, recall the definition of $c(\pi^*, \beta)$

$$\begin{aligned} c(\pi^*, \beta) &= E_{\pi^*,\beta} \left[\sum_{t=1}^{+\infty} c(X_t, A_t) \right] \\ &= \sum_{t=1}^{+\infty} \sum_{(x,a)\in\mathcal{K}'} c(x, a) \Pr[X_t = x, A_t = a]. \end{aligned}$$

We defined $c(x, a) = 0$ for each $x \neq \mathcal{S}$ and $c(\mathcal{S}, a_{\mathcal{S}}) = 1$. Therefore the expression simplifies to

$$c(\pi^*, \beta) = \sum_{t=1}^{+\infty} \Pr[X_t = \mathcal{S}, A_t = a_{\mathcal{S}}] = \rho(\mathcal{S}, a_{\mathcal{S}})$$

and this proves the second statement.$\square$

The theorem then shows that thanks to the way we defined the costs $c(x, a)$, by minimizing $c(\pi, \beta)$ we minimize the failure probability.

Remark: once the optimal policy is computed, it is important to consider that the temporal deadline D is met in expectation considering *all* executions, including those leading to a failure. The time of completion of a failing deployment is smaller because when a failure occurs the state enters $\mathcal{S}$ and then deterministically moves into T without accruing additional time (in other words, the state evolution takes a shortcut towards T via $\mathcal{S}$). Hence, if one just looks at the expected time to completion conditioned to a successful deployment this value may be higher than D and this mismatch grows as the failure probability grows. This is evident considering the following expression:

$$E[T_{dep}] = E[T_{dep}|Succ] \Pr[Succ] + E[T_{dep}|Fail] \Pr[Fail]$$

where T_{dep} is the deployment time and for what we just said $E[T_{dep}|Fail]$ is smaller than $E[T_{dep}|Succ]$. On the other hand, from a practical point of view one will select the size of the team so that the $\Pr[Fail]$ is small, and in such case $E[T_{dep}|Succ] \Pr[Succ]$ will converge towards $E[T_{dep}]$ and then satisfy the given deadline D.

A. The deployment algorithm

Based on the CMDP formulation, we can implement the following deployment algorithm that generalizes and improves the one we formerly proposed. The main tenet of this strategy is that no coordination is needed between agents, i.e., they do not communicate among each other, nor they know how many robots are in the team (i.e., they do not know the value of K.) This strategy not only serves as yardstick for comparison with more sophisticated coordination mechanisms to be developed in the future, but is also robust and easy to setup because team members can just be added without the need for any re-configuration or replanning. The pseudocode is sketched in Algorithm 1. Each robot picks a random target vertex in T (line 1), builds the associated CMDP (line 2) computes the optimal strategy π^* (line 3) and then navigates to v according to the strategy (line 4).

```
1 choose random vertex $v \in T$;
2 build CMDP instance $\mathcal{CMDP}$ with $\mathbf{M} = \{v\}$;
3 $\pi^* \leftarrow$ SolveCMDP($\mathcal{CMDP}$);
4 navigate to $v$ according to policy $\pi^*$;
```

Algorithm 1: Deployment algorithm implemented by each agent.

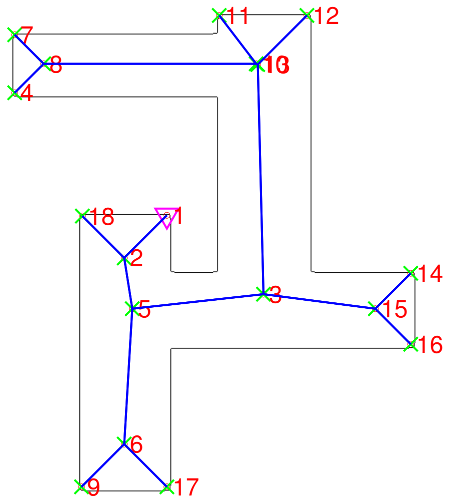
For sake of simplicity, in line 1 the target vertex $v \in T$ is chosen using a uniformly random distribution. Better approaches are possible, e.g., sending more robots to the more challenging locations, but we omit them for lack of space. Our approach is however independent from the process used to select v.

For a given instance of the deployment problem, we can then compute the failure probability for each of the N nodes in T, where the failure of a node is defined as the probability that a robot following the optimal policy will fail to reach the node. In [7] we derived a detailed analysis relating the probability of success given K, and a temporal deadline (see the referenced paper for details.) The analysis assumed knowledge of the failure probability we just described, but our former algorithm did not provide it, so we had to revert to an approximation. Thanks to the CMDP formulation, instead, we can now provide a sharper, exact bound.

V. Simulation results

In this section we test the proposed algorithm in two environments[4]. We on purpose use the same maps we used in [7] (see Figure 2). Note however that a direct comparison between the CMDP formulation and the one we proposed in [7] is not possible because the two solutions are subject to different temporal constraints. CMDPs produce policies that obey a constraint on the *expected* time to complete

[4]Matlab code implementing the simulations described in this section is available for download at `robotics.ucmerced.edu`.


(a) Simple map

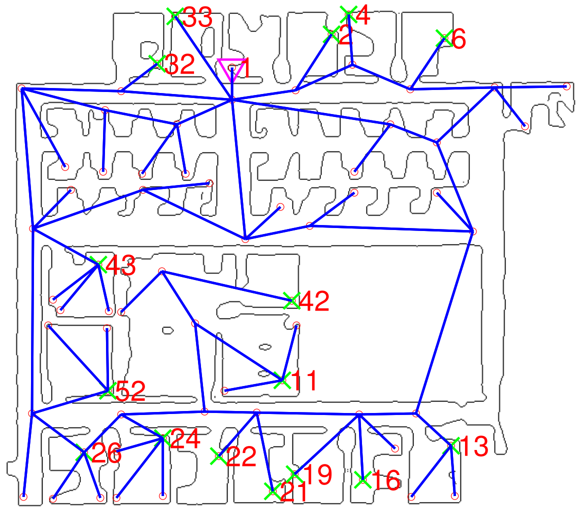

(b) Sdr40 map

Fig. 2: The two maps used to experimentally evaluate the deployment policies computed with a CMDP are the same we used in [7]. The deployment vertex is marked with a pink triangle, whereas goal vertices are indicated by green crosses. Edges between vertices indicate that a path exists. The left map is drawn by hand, whereas the right one is derived from the publicly available *Radish* dataset).

the deployment, whereas the algorithm we proposed in [7] obeys to a strict temporal deadline (without expectation).

In each of the map the graph is overlaid onto the blue print of the environment. Target vertices are marked with a number and the deployment vertex is indicated with a pink triangle (vertex number 1). Each edge is characterized with a different S function. Figure 3 shows the prototypical sigmoid function associated with each edge and providing the probability of success as a function of the time spent traversing the edge. Sigmoidal functions are commonly used when modeling risk [23]. Note however, that our findings are parametric with respect to S, as long as it satisfies the condition we gave in Section IV.

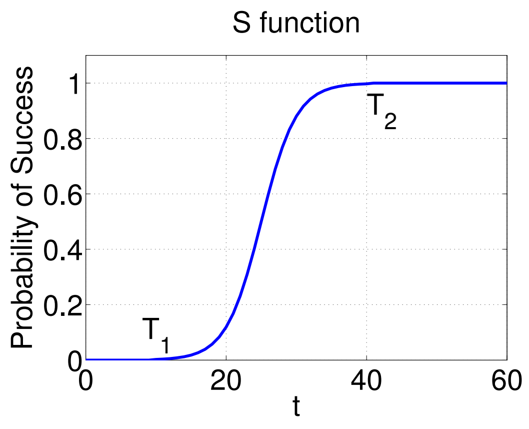


Fig. 3: General shape of the S function associated with every edge in the graph.

Parameters T_1 and T_2 are a function of the edge and in general different for each edge. In particular T_1 is function of the euclidean distance between the two vertices and T_2 is a function of the clearance on the path between the vertices (low clearance implies a higher value for T_2 because navigation is more challenging, so the robot needs to slow down to safely navigate).

Figure 4 and Figure 5 show the results of simulation deployment for policies obtained with the CMDP algorithm for different temporal deadlines. Based on these charts, one can then decide how many robots should allocated to the team in order to match a given probability of success while satisfying a constraint on the expected deployment time.

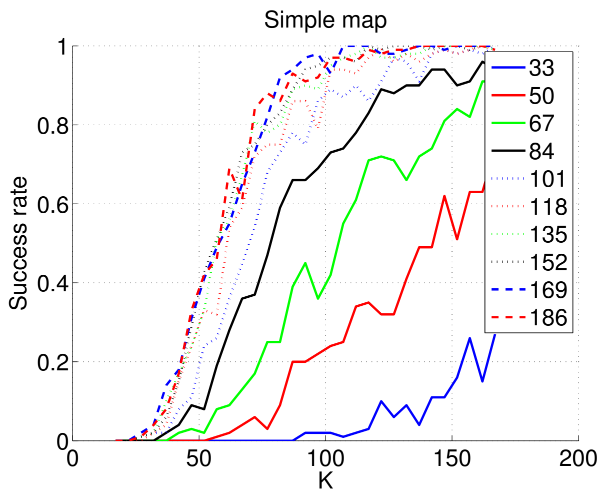


Fig. 4: Success rate as a function of the number of robots for different temporal deadlines for the simple map displayed in figure 2a.

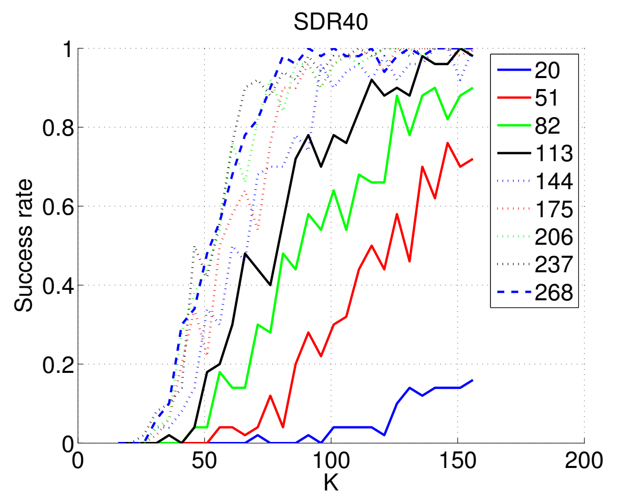


Fig. 5: Success rate as a function of the number of robots for different temporal deadlines for the sdr40 map displayed in figure 2b.

Finally in figure 6 we show the trends of the average time to completion for one specific case (sdrmap, temporal deadline $D = 113$), and we plot both the average for all runs (blue line) and the average restricted to the successful runs only (red line).

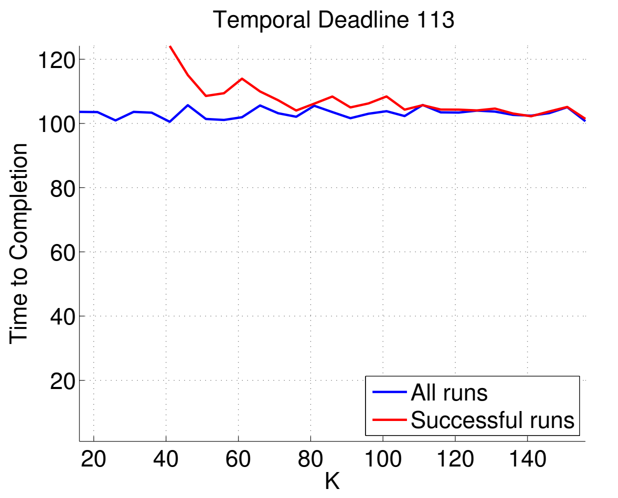


Fig. 6: Average time to completion of the task as a function of the swarm size for the sdr40 map and a temporal deadline of 113.

Experimental data confirm the explanation regarding the expected execution time we gave in the remark at the end of section IV. In expectation, the deployment always terminates within the given deadline (jitter in the figure is due to averaging a limited number of trials). For low values of K the failure probability is higher and if one looks at the expected time to completion restricted to the successful runs (red line), the expectation is higher than the deadline, and the constraint is met only when considering all runs (i.e., including successful and unsuccessful ones – blue line). As the number of agents deployed grows, the fraction of robots successfully completing the deployment grows too, and then the two expectations eventually converge. To put this chart into perspective, one should relate this chart to the trend of the line for $D = 113$ given in Figure 5. Given that from an operational point of view one wants to operate in a regime where the success probability is high, it follows that under these conditions our model produces deployment strategies minimizing risk and meeting the temporal deadline.

Given that the problem we study is novel, a comparison with alternative methods is not immediate. One could compare with the method we presented in [7]. While for simple problems the solution could be comparable in terms of performance, the method in [7] does not provide an exact error bound, so it is less informative when it comes to decide how many agents to should be included in the team.

Alternatively, one could use algorithms searching for shortest paths on graphs, but they consider just one metric, whereas our method allows, in a principled way, to minimize one cost (failure probability) while keeping a bound on another cost (completion time).

VI. Conclusions and Future Work

In this paper we have studied the problem of rapid deployment of multiple robots under temporal constraints using the theory of CMDPs. CMDPs offer a rich framework for further work and generalizations. First, there exist numerous results for CMDPs where the action state $A(x)$ is a compact set rather than a finite set. This extension allows to remove the discretization on the actions set. Constraints can also be introduced to model other limited resources, like for example to allow for communications between robots but imposing costs to keep them to a minimum or to rely on communication only when it is highly likely to succeed (for example when robots are close to each other). Communication would enable replanning, for example to reroute robots to a different target location when they learn it has already been reached by other robots.

The optimal policy π^* depends on β, and β in turn does not need to be concentrated on a single state d but may be a general mass distribution over $\mathbf{X}$. In certain situations there may be the possibility to choose where to deploy the robots before the deployment starts. By studying the relations between the failure probability of π^* and β it may be possible to pick the deployment vertex, or a probability distribution over a set of deployment vertices, giving the lowest failure probability. Moreover, paralleling CMDPSs, there exist a theory of Constrained Partially Observable Markov Decision Processese [8], [14] that we intend to study to account for uncertainties in the localization process or in sensing in general.

References

[1] E. Altman. Constrained Markov decision processes with total cost criteria: Occupation measures and primal LP. *Mathematical methods of operations research*, 43:45–72, 1996.

[2] E. Altman. *Constrained Markov Decision Processes*. Stochastic modeling. Chapman & Hall/CRC, 1999.

[3] M. Batalin and G. Sukhatme. The Design and Analysis of an Efficient Local Algorithm for Coverage and Exploration Based on Sensor Network Deployment. *IEEE Transactions on Robotics*, 23(4):661–675, Aug 2007.

[4] D. P. Bertsekas. *Dynamic Programming & Optimal Control*, volume 1 and 2. Athena Scientific, 2005.

[5] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: from Natural to Artificial Systems*, volume 4. Oxford University Press New York, 1999.

[6] F. Bullo, J. Cortés, and S. Martínez. *Distributed Control of Robotic Networks*. Princeton, 2009.

[7] S. Carpin, T.H. Chung, and B. Sadler. Theoretical foundations of high-speed robot team deployment. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2025–2032, 2013.

[8] R.C. Chen, K. Wagner, and G.L. Blankenship. Constrained partially observable Markov decision processes with probabilistic criteria for adaptive sequential detection. *IEEE Transactions on Automatic Control*, 58(2):487–493, 2013.

[9] J. Cortés, S. Martínez, T. Karatas, and F. Bullo. Coverage control for mobile sensing netorks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004.

[10] X. Ding, M. Kloetzer, Y. Chen, and C. Belta. Automatic Deployment of Robotic Teams. *IEEE Robotics & Automation Magazine*, 18(3):75–86, 2011.

[11] X. Ding, A. Pinto, and A. Surana. Strategic Planning under Uncertainties via Constrained Markov Decision Processes. In *IEEE International Conference on Robotics and Automation*, pages 4568–4575. IEEE, 2013.

[12] J. Fink, A. Ribeiro, and V. Kumar. Robust Control of Mobility and Communications in Autonomous Robot Teams. *IEEE Access*, 1:290–309, 2013.

[13] H. Huang, G.M. Hoffmann, S.L. Waslander, and C.J. Tomlin. Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3277–3282, 2009.

[14] D. Kim, J. Lee, K.-E. Kim, and P. Poupart. Point-based value iteration for constrained POMDP. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 1968–1974, 2011.

[15] M. Kloetzer and C. Belta. Temporal Logic Planning and Control of Robotic Swarms by Hierarchical Abstractions. *IEEE Transactions on Robotics*, 23(2):320–330, 2007.

[16] A. Kolling and S. Carpin. Extracting surveillance graphs from robot maps. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2323–2328, 2008.

[17] L. Matignon, L. Jeanpierre, and A.-I. Mouaddib. Coordinated multi-robot exploration under communication constraints using decentralized Markov decidion processes. In *Proceedings of the twenty-sixth AAAI conference on artificial intelligence*, pages 2017–2023, 2012.

[18] D. Mellinger, N. Michael, and V. Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. In *Proceedings of the International Symposium on Experimental Robotics*, Dec 2010.

[19] R. Morlok and M. Gini. Dispersing robots in an unknown environment. In *7th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2004.

[20] J.L. Pearce, P.E. Rybski, S.S. Stoeter, and N.P. Papanilolopoulos. Dispersion behaviors for a team of multiple miniature robots. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 1158–1163, 2003.

[21] A. Purohit and P. Zhang. Controlled-mobile Sensing Simulator for Indoor Fire Monitoring. In *Wireless Communications and Mobile Computing Conference*, pages 1124–1129, 2011.

[22] A. Purohit, P. Zhang, B. Sadler, and S. Carpin. Deployment of swarms of micro-aerial vehicles: from theory to practice. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2014 (accepted for publication).

[23] M. Rausand and A. Høyland. *System Reliability Theory: Models, Statistical Methods, and Applications*, volume 396. John Wiley & Sons, 2004.

[24] M. Schwager, J. McLurkin, and D. Rus. Distributed Coverage Control with Sensory Feedback for Networked Robots. In *Proceedings of Robotics: Science and Systems*, 2006.