Mixed Observable RRT: Multi-Agent Mission Planning in Partially Observable Environments

Kasper Johansson[1], Ugo Rosolia[2], Wyatt Ubellacker[2], Andrew Singletary[2], and Aaron D. Ames[2]

***Abstract*— This paper considers centralized mission-planning for a heterogeneous multi-agent system with the aim of locating a hidden target. We propose a mixed observable setting, consisting of a fully observable state-space and a partially observable environment, using a hidden Markov model. First, we construct rapidly exploring random trees (RRTs) to introduce the mixed observable RRT for finding plausible mission plans giving way-points for each agent. Leveraging this construction, we present a path-selection strategy based on a dynamic programming approach, which accounts for the uncertainty from partial observations and minimizes the expected cost. Finally, we combine the high-level plan with model predictive controllers to evaluate the approach on an experimental setup consisting of a quadruped robot and a drone. It is shown that agents are able to make intelligent decisions to explore the area efficiently and to locate the target through collaborative actions.**

I. Introduction

Planning under uncertainty is important for autonomous systems. Similar to how humans make decisions based on what they observe—for instance, when driving on a busy road or playing sports—autonomous robotic systems must be able to act based on observations from the environment. Exploration tasks are one type of problem where robots have partial knowledge about the environment complemented with observations along the mission [1]. The seems to exist no satisfying solution to such exploration when multiple robots are to cooperate and act simultaneously.

The literature on environment mappings for navigation and planning is rich [1]–[4]. Exploration can be done using a policy that maps the system's state to a control action [5]. The computational burden associated with planning over policies has been extensively studied [6]–[12], and can be reduced when a system's state can be perfectly measured [5], [6], [11], [13]. Tube model predictive control (MPC) strategies involve another type of feedback policies [7]–[10]. Strategies where only partial state knowledge is available have been studied in [14]–[16]. In [5] the authors consider multi-modal measurement noise by planning over a tree of trajectories where each branch is associated with a unique set of observations made during a mission.

A popular tool for path-planning is the rapidly exploring random tree (RRT), which is a sampling based search algorithm [17]. RRTs have been leveraged for path-planning in various problems, like navigating among moving obstacles [18], kinodynamic motion-planning [19], and probabilistically robust path-planning [20], among many others. The RRT* algorithm is an extension to the RRT [21], and it has been shown that RRT* converges to the optimal path [22].

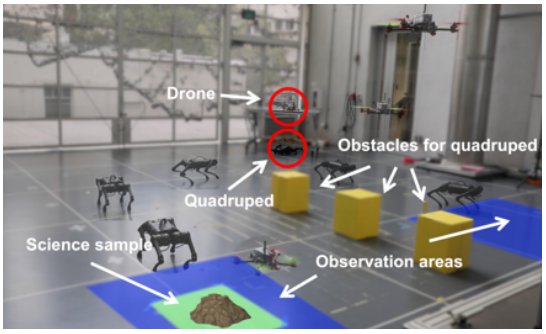


Fig. 1: A quadruped and a drone are initiated inside the red circles. They locate a science sample by sharing observations from the environment.

In this work we introduce a novel approach to path-planning in partially observable environments for multi-agent systems. Agents act simultaneously to locate a target, as illustrated in Figure 1. We leverage RRTs to generate a sample of event-based plans, where a plan consists of a sequence of way-points that changes as a function of the observations agents make. The best plan is found by minimizing an expected cost over the tree of plans, as in [5]. To allow a multi-agent system to follow the high-level plan we leverage MPC algorithms. The contribution is threefold:

- First, we model a cooperative multi-agent exploration mission using a mixed observable setting (Section II). Based on this formalism, we modify the standard RRT algorithm to generate a sample of plans in a discrete partially observable environment (Section III). This approach defines the *mixed observable RRT (MORRT)*.
- Next, we introduce a dynamic program for finding the plan that minimizes the expected cost (Section III).
- Finally, we demonstrate our approach experimentally by integrating the high-level plan, consisting of way-points for a heterogeneous multi-agent system, with local MPC algorithms (Section IV).

Notation: For a vector $b \in \mathbb{R}^n$, $n \geq 2$, and an integer $s \in \{1, \ldots, n\}$, denote $b[s]$ as the sth component of b and $b^\top$ as its transpose. $\text{diag}(b)$ is a diagonal matrix with elements b. Let $\mathbb{N} = \{0, 1, \ldots\}$ and $\mathbb{R}_{0+} = [0, \infty)$. Furthermore, given a set $\mathcal{Z}$, $|\mathcal{Z}|$ denotes its number of elements. For an integer k, we denote the kth Cartesian product as $\mathcal{Z}^k = \mathcal{Z} \times \ldots \times \mathcal{Z}$. For two sets $\mathcal{Z}_1, \mathcal{Z}_2$ we define the outer product $\mathcal{Z}_1 \otimes \mathcal{Z}_2 = \{(z_1, z_2) : z_1 \in \mathcal{Z}_1, z_2 \in \mathcal{Z}_2\}$. The $N \times N$ identity matrix is given by $I_{N\times N}$. Lastly, the indicator function $\mathbb{1}[y_1 = y_2] = 1$ if $y_1 = y_2$ and 0 otherwise.

[1]KTH Royal Institute of Technology, Stockholm, Sweden. kasperjo@kth.se
[2]California Institute of Technology, Pasadena, USA. {urosolia, wubellac, asinglet, ames}@caltech.edu

II. Problem Formulation

First we introduce the multi-agent system, and describe the discrete partially observable environment and its interaction with the system. Then, we introduce the control objectives.

A. System and Environment Models

Consider a multi-agent system of M agents with the aim of locating a target that is hidden in $\mathbb{R}^2$. Agents have some prior knowledge of where the target is located, and know that it is at one of a finite number of known *goal nodes*, i.e., points in $\mathbb{R}^2$. As an example, see Figure 1 where a quadruped and a drone are locating a science sample. In order to find the hidden target, the agents make observations in *observation areas*, i.e., certain regions of the state-space. Observations are shared between the agents.

We introduce a mission planner that has to compute a set of way-points for the multi-agent system. Let $\mathcal{X}^a \subseteq \mathbb{R}^2$ be the state-space of agent $a \in \{1, \dots, M\}$, and $\mathcal{X}^a_{\text{obs}} \subseteq \mathbb{R}^2$ its obstacle region. Denote the space where agent a is free to move by $\mathcal{X}^a_{\text{free}} = \mathcal{X}^a \setminus \mathcal{X}^a_{\text{obs}}$. The way-point x^a_{k+1} is computed as

$$x^a_{k+1} = x^a_k + \Delta x^a_k, \tag{1}$$

where $\Delta x^a_k \in \mathbb{R}^2$ is the way-point change at time k. We represent the state of the multi-agent system as $x_k = [x^1_k, ..., x^M_k]$.

As the exact location of the target is unknown, the planner has to compute way-points based on partial observations which are used to update the belief about the environment. The planner computes way-points for each agent to minimize a cost, which is a function over the agent's state, the way-point, and the goal nodes, as described further down. The agents' belief of where the target is located is modelled as a partially observable environment. There are a finite number, G, of goal nodes where the target can be located, corresponding to G different environment states. In Figure 1 $G = 2$, corresponding to points inside of the blue squares. The discrete environment evolution is modeled using a hidden Markov model (HMM) [23] given by the tuple $\mathcal{H} = (\mathcal{E}, \mathcal{O}, T, Z)$:

- $\mathcal{E} = \{1, \dots, G\}$ is a set of partially observable environment states.
- $\mathcal{O} = \{1, \dots, G\}$ is the set of observations for the partially observable state $e \in \mathcal{E}$.
- The function $T : \mathcal{E} \times \mathcal{E} \times \mathbb{R}^n \to [0, 1]$ describes the probability of transitioning to a state e' given the current environment state e and system's state x—i.e.,

$$T(e', e, x) := \mathbb{P}(e'|e, x).$$

- The function $Z : \mathcal{E} \times \mathcal{O} \times \mathbb{R}^n \to [0, 1]$ describes the probability of observing o, given the current environment state e and the system's state x—i.e.,

$$Z(e, o, x) := \mathbb{P}(o|e, x).$$

We define the trajectory $\mathbf{x}^a_k = [x^a_1, \dots, x^a_k]$, where $x^a_\ell \in \mathcal{X}^a_{\text{free}}$, $\forall \ell \in \{1, \dots, k\}$, $k \geq 1$. Let $j(k) : \mathbb{N} \to \mathbb{N}$ denote the number of observations made until time k, and let $\mathbf{o}_{j(k)} = [o_0, \dots, o_{j(k)-1}]$, $j(k) \geq 1$, with $\mathbf{o}_0 = \emptyset$ be the corresponding vector of observations. We introduce the belief vector $b_k \in \mathcal{B} = \{b \in \mathbb{R}^{|\mathcal{E}|}_{0+} : \sum_{i=1}^{|\mathcal{E}|} b[e] = 1\}$, which is a sufficient statistics [23]. Here, entry $b_k[e]$ represents the posterior probability, at time k, that the system's state e_k equals to $e \in \mathcal{E}$, given the vector of observations $\mathbf{o}_{j(k)}$, the collection of agent trajectories $\mathbf{x}_k = \{\mathbf{x}^a_k\}_{a=1}^M$, the initial state $x(0) = [x^1(0), .., x^M(0)]$, and the initial belief vector $b(0)$ at time $t = 0$—i.e., $b_k[e] = \mathbb{P}(e|\mathbf{o}_{j(k)}, \mathbf{x}_k, x(0), b(0))$.

As mentioned, agents make observations in certain regions of the state-space, observation areas, to update the belief of where the target is located. When an observation is made, the mission plan branches into $|\mathcal{O}|$ sub-paths for each agent, where each sub-path corresponds to one observation, as shown in Figure 2.

B. Control Objectives

The objective is to find a mission plan for the system to act in the uncertain environment. During a mission, agents will make observations, and the executed mission depends on what observations are made—i.e., the realization of the stochastic observations. For this reason the mission plan is a policy, which depends on a fixed number of possible realizations. In other words the sequence of way-points changes as a function of the observations. For instance, in Figure 2, there are three possible paths for the agent to take, corresponding to observing $[0, 0]$, $[0, 1]$, and $[1]$, respectively.

Let $N_{\max}$ denote the maximum length of the mission. For a mission starting at time $t = 0$ and ending at time $N \leq N_{\max}$, we denote the policy for agent a by $\boldsymbol{\pi}^a = [\pi^a_0, \dots, \pi^a_{N_{\max}-1}]$, where at each time k the policy $\pi^a_k : \mathcal{O}^{j(k)} \times (\mathbb{R}^{2\times M})^k \times \mathcal{B} \to \mathbb{R}^2$ maps the environment observations $\mathbf{o}_{j(k)}$ up to time k, the system's trajectory $\mathbf{x}_k$, the initial state $x(t)$, and the initial belief $b(t)$ to the way-point Δx^a_k—i.e., $\Delta x^a_k = \pi^a_k(\mathbf{o}_{j(k)}, \mathbf{x}_k, x(t), b(t))$,

Let $p = [p^1, \dots, p^M]$ denote M paths, one for each agent a, where $p^a = [x^a_0, \dots, x^a_{N_p}]$ denotes the path of agent a, for a mission of length N_p. We denote the set of possible paths, for all agents, by $\mathcal{P}$. Each $p = [p^1, \dots, p^M] \in \mathcal{P}$ corresponds to a set of paths for each agent, associated with one realization of observations along a mission. Hence, $\mathcal{P}$ is a mapping from the stochastic observations to the realized paths. The length of the paths $p^a \in p$ is denoted by N_p, and $N_{\max} = \max(N_p)$ such that $p \in \mathcal{P}$. Given the initial state $x(t) = [x^1(t), \dots, x^M(t)]$ and environment belief $b(t)$ starting at time $t = 0$, as well as a plan $\boldsymbol{\pi} = \{\boldsymbol{\pi}^a\}_{a=1}^M$, we define the cost of the policy for agent a as follows:

$$J^a(x(t), b(t)) = \mathbb{E}_{\mathcal{P}}\left[\sum_{k=0}^{N_p-1} h(x^a_k, \Delta x^a_k, e_k) + h_N(x^a_N, e_N)\middle| b(t)\right], \tag{2}$$

where the stage cost $h : \mathbb{R}^2 \times \mathbb{R}^2 \times \mathcal{E} \to \mathbb{R}$ and terminal cost $h_N : \mathbb{R}^2 \times \mathcal{E} \to \mathbb{R}$ are functions of the partially observable environment state $e \in \mathcal{E}$, and the expectation is over the stochastic realizations of agent paths $\mathcal{P}$, where each realization has a corresponding path length N_{p}.

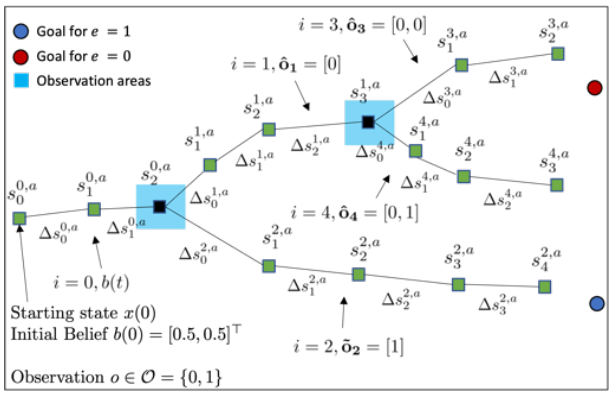


Fig. 2: An MORRT plan for an agent a. Nodes and edges correspond to way-points and way-point changes, respectively, and each branch i is associated with a unique observation vector $\hat{\mathbf{o}}_i$. For this plan $\mathcal{I} = \{0, 1, 2, 3, 4\}$, corresponding to five branches, where $\mathtt{P}(3) = \mathtt{P}(4) = 1$, $\mathtt{P}(1) = \mathtt{P}(2) = 0$, $\mathtt{O}(0) = \mathtt{O}(1) = 1$, and $\mathtt{O}(2) = \mathtt{O}(3) = \mathtt{O}(4) = 0$. Furthermore, branch 0 corresponds to $b(t)$, and branches 1-4 correspond to the vectors of observations $\hat{\mathbf{o}}_1 = [0]$, $\hat{\mathbf{o}}_2 = [1]$, $\hat{\mathbf{o}}_3 = [0, 0]$, and $\hat{\mathbf{o}}_4 = [0, 1]$, respectively.

III. The Mixed Observable RRT

We now introduce the mixed observable RRT (MORRT). First, we reformulate the cost (2) as an optimization problem over the trees of way-points for all agents. Secondly, we describe how to expand the MORRT. Finally, we explain how the best plan is found.

A. Cost Reformulation

The objective is first to reformulate the cost (2) as a summation over system states. We introduce the set of branches, $\mathcal{I}$, in a plan, and denote $\mathcal{I}_0 = \mathcal{I} \backslash \{0\}$. Branch $i = 0$ corresponds to the initial belief $b(t)$, and each remaining branch $i \in \mathcal{I}_0$ corresponds to a unique observation vector $\hat{\mathbf{o}}_i = [o_0, ..., o_{j-1}]$, for some integer $j \in \mathbb{N}$ of observations made until this branch. Each branch has $N_i + 1$ nodes, corresponding to system states $[s_0^i, ..., s_{N_i}^i]$, where the last node of a branch is the same as the first node of the child branches. We use $[s_0^{i,a}, ..., s_{N_i}^{i,a}]$ to denote the states of agent a in branch i. Note that s_ℓ^i denotes the system states in branch i where $\ell = 0$ at the root of this branch, which is equivalent to the system state x_k for some k, where $k = 0$ at the root of branch 0. Moreover, let the function $\mathtt{P}(i)$ return the parent branch of branch i, and let $\mathtt{O}(i) = 1$ if branch i ends at an observation node, and $\mathtt{O}(i) = 0$ otherwise.

Figure 2 shows the plan for an agent a. The agent starts at $s_0^0 = x(0)$ in the first branch $i = 0$. An observation is made at the third node of this branch, and the plan branches into $i = 1, 2$. If the agent observes $e = 0$ in the first branch it follows branch 1 and makes another observation at the fourth node of this branch; but, if it observes $e = 1$ it goes straight to the blue goal node without making another observation (branch 2). Black observation nodes, inside of observation areas, are both the final node of one branch and initial nodes of the branches originating from the observation node; for instance, $s_2^{0,a} = s_0^{1,a} = s_0^{2,a}$, but to make the figure less cluttered, this is not illustrated.

Following the exact solution from [5] the expectation (2) can be written, equivalently, as a summation over way-points. In [5] an observation is made, and the plan branches, at each time step k. However, we only allow observations in observation areas, so a plan branches at the beginning of each branch $i \in \mathcal{I}_0$.

Let $\Theta(o_k, x_k) = \mathrm{diag}\big(Z(1, o_k, x_k) \quad \dots \quad Z(|\mathcal{E}|, o_k, x_k)\big)$, and make the assumption that the environment state stays fixed—i.e., the target does not move—meaning that $T(e_i, e_j, x_k) = 1$ if $i = j$ and $T(e_i, e_j, x_k) = 0$ otherwise.

Similar to [5], the cost (2) can be rewritten as

$$\begin{aligned} &J^a(\mathcal{I}, x(t), b(t)) \\ &\quad = \sum_{i \in \mathcal{I}} \Bigg\{ \sum_{\ell=0}^{N_i - 1} \sum_{e \in \mathcal{E}} v^i[e] h(s_\ell^{i,a}, \Delta s_\ell^{i,a}, e) \\ &\qquad + \mathbb{1}[\mathtt{O}(i) = 0] \sum_{e \in \mathcal{E}} v^i[e] h_N(s_{N_i}^{i,a}, e) \Bigg\}, \end{aligned} \tag{3}$$

where o_i is the observation made at the beginning of branch i, $s_0^0 = x(t)$, and $v^i[e] = \Theta(o_i, s_0^i) v^{\mathtt{P}(i)}[e]$ is the unnormalized belief. For more details please refer to [5].

Problem 1. *For a system of M agents our goal is to solve the following finite-time optimal control problem:*

$$\begin{aligned} J^*(\mathcal{I}, x(t), b(t)) = \min_{\mathcal{I}, \mathbf{\Delta s}} \quad & \sum_{a=1}^{M} J^a(\mathcal{I}, x(t), b(t)) \\ \text{s.t.} \quad & s_{\ell+1}^{i,a} = s_\ell^{i,a} + \Delta s_\ell^{i,a}, \forall i \in \mathcal{I} \\ & s_0^{i,a} = s_{N_{\mathtt{P}(i)}}^{\mathtt{P}(i),a}, \forall i \in \mathcal{I}_0 \\ & s_0^{0,a} = x^a(0), v_0^0 = b(0), \\ & v^i[e] = \Theta(o_i, s_0^i) v^{\mathtt{P}(i)}[e], \forall i \in \mathcal{I}_0, \\ & s_{\ell+1}^{i,a} \in \mathcal{X}_{free}^a, \forall i \in \mathcal{I}, \\ & \forall a \in \{1, \dots, M\}, \\ & \forall \ell \in \{0, \dots, N_i - 1\}, \end{aligned} \tag{4}$$

where $\mathbf{\Delta s} = \{\mathbf{\Delta s}^a\}_{a=1}^M$, and each $\mathbf{\Delta s}^a$ is a matrix of way-point changes for agent a:

$$\mathbf{\Delta s}^a = [\Delta s_0^{0,a}, \dots, \Delta s_{N_0-1}^{0,a}, \dots, \Delta s_0^{I,a}, \dots, \Delta s_{N_I-1}^{I,a}],$$

where $I = |\mathcal{I}| - 1$. The above optimization is over both the set of branches $\mathcal{I}$, and the matrices Δs^a for each agent.

B. Building the MORRT

We define the function $\mathtt{Extend}_{RRT}(\mathcal{T}, x_{\text{rand}}, \Delta x)$, which extends a tree $\mathcal{T}$ by adding a node a distance Δx from the nearest node $x_{\text{near}} \in \mathcal{T}$ on the straight line between x_{near} and x_{rand}. The RRT* extension [24] can be leveraged by introducing the *cost heuristic* for a path as the expected cost (2) given that the path is taken regardless of what observations are made.

The *mixed observable RRT (MORRT)* is expanded as follows: an observation is recorded when a node is placed inside an observation area; nodes are added to the tree until a predetermined number of nodes, *observation nodes*, have

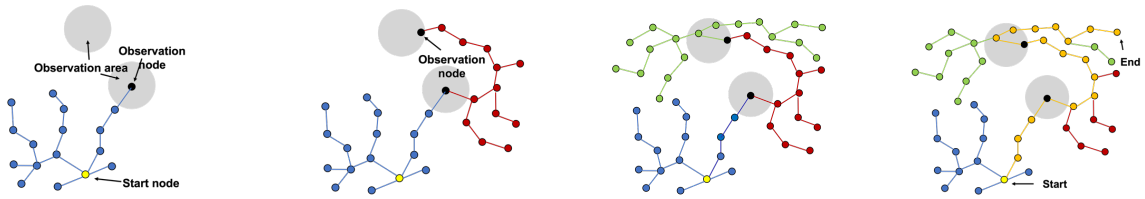


Fig. 3: The MORRT expansion for $N_{\text{obs}} = 1$. The tree expands from left to right. The yellow path to the right shows one possible path in the MORRT.

been placed inside observation areas; and new RRTs are initialized from all observation nodes. We call this search algorithm the . For simplicity, only one observation is allowed in each observation area for a single plan. Algorithm 1 illustrates the multi-agent RRT. A tree $\mathcal{T}_a$ is initiated for each agent a (line 2). If there exists unexplored observation areas in the environment—meaning that no nodes have been placed in these by other RRTs—the tree expands until some number, N_{obs}, of observation nodes have been placed inside unexplored observation areas (lines 3-6). The observation nodes can be placed in any observation area that is still unobserved—i.e., multiple observation nodes may end up in the same observation area as long as they are from the same RRT. If a node is placed in an observation area that contains nodes from other RRTs, no observation is recorded. Moreover, we only allow one observation per path in a single tree; if two nodes along the same path of the same tree are placed inside observation areas, an observation is only recorded for the first node.

If all observation areas are explored, the tree $\mathcal{T}_a$ is expanded until its number of nodes equals a constant K (lines 7-10). The multi-agent RRT is defined as the set of all expanded trees (line 11). Its set of observation nodes, RRT.o, is initialized as the outer product of all agents' observation nodes (line 12). Each observation node n_o^a from an agent tree $\mathcal{T}_a$ corresponds to a path ending at n_o^a. The `shortenPaths` function (line 13) modifies the observation nodes as follows: for each set of observation nodes $\mathcal{N} \in$ RRT.o, pick the node $n_o^a \in \mathcal{N}$ corresponding to the shortest path of length L, say; for all other nodes, $\mathcal{N}_0 \in$ RRT.$o \setminus \{n_o^a\}$, replace them with a node higher up in the corresponding path, such that this node corresponds to a path of length L. The nodes in every set $\mathcal{N} \in$ RRT.o will then correspond to paths of the same length.

Algorithm 1 RRT$\leftarrow$ `RRT`$(M, x_{\text{init}}, N_{\text{obs}}, K, \Delta x)$

1: **for** $a = 1$, a++, **while** $a <= M$ **do**
2: $\quad\mathcal{T}_a$.`init`$(x_{\text{init}}[a])$
3: $\quad$**if** `observationAreas()` **then**
4: $\quad\quad$**while** $\mathcal{T}_a.n_{\text{obs}} < N_{\text{obs}}$ **do**
5: $\quad\quad\quad x_{\text{rand}} \leftarrow$ `RandomState()`
6: $\quad\quad\quad$`Extend`$_{RRT}(\mathcal{T}_a, x_{\text{rand}}, \Delta x)$
7: $\quad$**else**
8: $\quad\quad$**while** $\mathcal{T}_a.n_{\text{nodes}} < K$ **do**
9: $\quad\quad\quad x_{\text{rand}} \leftarrow$ `RandomState()`
10: $\quad\quad\quad$`Extend`$_{RRT}(\mathcal{T}_a, x_{\text{rand}}, \Delta x)$
11: RRT $\leftarrow [\mathcal{T}_1, ..., \mathcal{T}_M]$
12: RRT.$o \leftarrow \mathcal{T}_1.o \otimes \mathcal{T}_2.o \otimes \cdots \otimes \mathcal{T}_M.o$
13: RRT.$o \leftarrow$`shortenPaths`(RRT.o)

This set is defined as an observation node for the system. To reduce computational time, we choose to only keep the nodes corresponding to paths with lowest cost heuristic to each observation area and agent (unless it is a single-agent system, for which we keep all nodes), before taking the outer product (line 12).

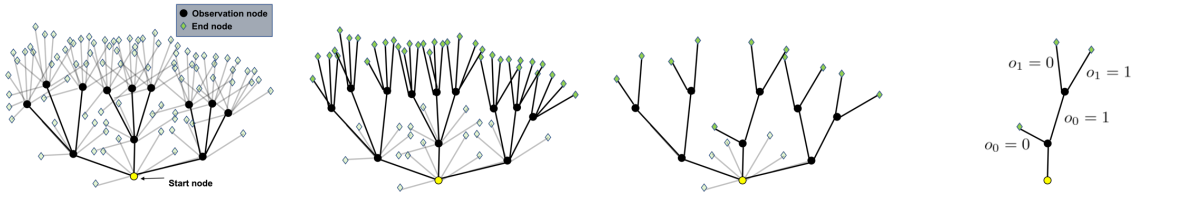
For each RRT that contains observation nodes, a child RRT, which expands independently of the parent RRT, is initialized from each of the N_{obs} observation nodes. All paths going through the child RRT include the path that originates from the parent's start node, going through the observation node. Algorithm 2 implements the MORRT leveraging Algorithm 1. In line 1 a "root RRT" is expanded. This RRT is stored in an array of parent RRTs (line 2). A child RRT is initialized from each observation node of the parent RRT. The procedure is repeated for all initialized RRTs until all observation areas are explored (lines 3-8).

An MORRT for one agent is illustrated in Figure 3 for $N_{\text{obs}} = 1$. A blue tree is initialized at the yellow start node, and expanded until it reaches one of the observation areas. The observation node is the start node for the red tree, which is expanded analogously. When the red tree reaches the second observation area, the green tree is initialized and expanded until some fixed size, since all observation areas have been visited by other RRTs. Any path that ends at a node that has no children is part of a possible plan (a plan does not have to include all three trees). One possible path is illustrated in yellow. Finally, note that the three trees would in general have crossing paths, since they are expanded independently of each other.

The MORRT generates a tree of RRTs as illustrated, for one agent, in Figure 4a. Here, the edges correspond to paths of a single RRT, and the nodes represent observation nodes connecting separately expanded RRTs. The connecting edges (or paths) are highlighted while other edges are faded. Nodes that do not have children—end nodes—are illustrated by green squares. In the figure each RRT has three observation

Algorithm 2 RRT$_{\text{root}} \leftarrow$ `MORRT`$(M, x_{\text{init}}, N_{\text{obs}}, K, \Delta x)$

1: RRT$_{\text{root}} \leftarrow$ `RRT`$(M, x_{\text{init}}, N_{\text{obs}}, K, \Delta x)$ `//ALGO 1`
2: parents = [RRT$_{\text{root}}$]
3: **while** parents **is not empty do**
4: $\quad$RRT$_{\text{parent}} \leftarrow$ parents[0]
5: $\quad$**for** x_{obs} **in** RRT$_{\text{parent}}$.o **do**
6: $\quad\quad$RRT$_{\text{child}} \leftarrow$ `RRT`$(M, x_{\text{obs}}, N_{\text{obs}}, K, \Delta x)$ `//ALGO 1`
7: $\quad\quad$parents.`append`(RRT$_{\text{child}}$)
8: $\quad$parents.`remove`(RRT$_{\text{parent}}$)



(a) Full MORRT. **(b)** Best plans, depth $h = 2$. **(c)** Best plans, depth $h = 1$. **(d)** Best plan (depth $h = 0$).

Fig. 4: Dynamic program for $N_{\text{obs}} = 3$ and $|\mathcal{O}| = 2$. The dynamic program works backward from dept $h = 2$ to depth $h = 0$ to find the best plan.

Algorithm 3 plan ← `bestPlan`(RRT)

```
1: h = RRT.depth
2: if not RRT.children then
3:     for o_{h-1} ∈ O^{h-1} do
4:         for o ∈ O do
5:             //Find best path
6: else
7:     for child in RRT.children do
8:         bestPlan(child)
9:     for o_{h-1} ∈ O^{h-1} do
10:        for o ∈ O do
11:            //Choose whether or not to branch
12: //Return plan with lowest expected cost
```

nodes, implying that $N_{\text{obs}} = 3$. Moreover, the tree of RRTs is two levels deep, so there are two observation areas (one observation area is explored at each observation node, and different branches are independent of each other and represent separate plans).

C. Finding the Best Plan From the MORRT

We leverage a dynamic program to find the best plan from the MORRT. The dynamic program is shown in Algorithm 3, and illustrated on an example with $N_{\text{obs}} = 3$ and $|\mathcal{O}| = 2$ (Figure 4). Let h denote an RRTs depth in the tree of RRTs, letting the root RRT have depth 0. The algorithm is initialized for the root RRT—`bestPlan`(RRT_{root})—and explores the tree of RRTs in depth first manner, so RRTs without children are considered first (line 2). These are the faded trees at depth $h = 2$ in Figure 4a. At $h = 2$, for each observation node, a plan of $|\mathcal{O}|^h$ paths—one path for each observation $o \in \mathcal{O}$ and all possible combinations of observations $o_{h-1} \in \mathcal{O}^{h-1}$ up to this point—is computed (lines 3-5). These plans are illustrated by the black lines, from depth $h = 2$, in Figure 4b; for each observation node at depth two, the four black lines represent the best path to take given $(o_0, o_1) = (0, 0), (0, 1), (1, 0)$, and $(1, 1)$, respectively, from that point in space.

Next, we consider the RRTs one step higher up in the tree of RRTs—$h = 1$ in the example—and proceed similarly (lines 9-11). For each o_{h-1}, the algorithm considers all $o \in \mathcal{O}$ and decides both whether or not the plan should branch (make another observation and continue in a child RRT, or neglect the child RRT), and the best path to take given that it branches or does not branch. If it is beneficial to branch, the best branch is computed for the set of observations o_{h-1}, and the corresponding best plan from the child RRT, from the previous step, is remembered. In Figure 4c, the plan fully branches the left- and right most trees, and only branches one path in the middle tree. Note that there are now only two paths originating from depth $h = 2$.

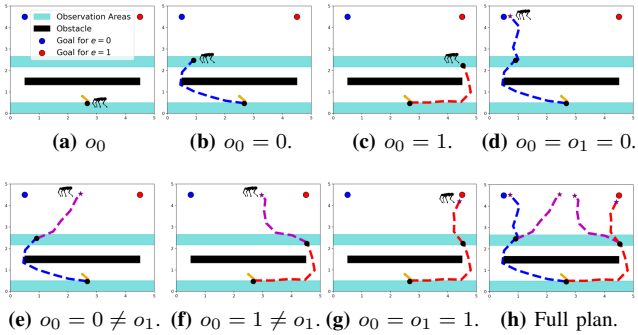
This continues until depth $h = 0$, where a plan is returned (Figure 4d). The plan is executed as follows: the agent starts at the yellow node. It makes an observation o_0. If $o_0 = 0$, the agent goes left and ends the mission; if $o_0 = 1$ the agent makes another observation o_1 and takes one of two paths.

IV. Results

Let us now evaluate the MORRT by considering simulations of both a single-agent and a multi-agent system, as well as a hardware experiment using a quadruped and a drone. The implementation code is available online.[1]

A. Single-Agent Simulation

The single-agent simulation is shown in Figure 5. A quadruped is initiated below the dark obstacle, and looks for a science sample that is located either in the top left ($e = 0$) or in the top right ($e = 1$). The initial belief is that the sample is in either of the top corners with equal probability: $b(0) = [0.5, 0.5]$. Observations are made in the blue regions, and are correct with 80% probability. The quadruped first visits the bottom observation area (Figure 5a). If it observes $o_0 = 0$ it follows the blue path to the left of the obstacle (Figure 5b), but if $o_0 = 1$ it follows the red path to the right (Figure 5c). Figures 5d and 5e show the plan from the second observation area, given that the first observation is $o_0 = 0$;

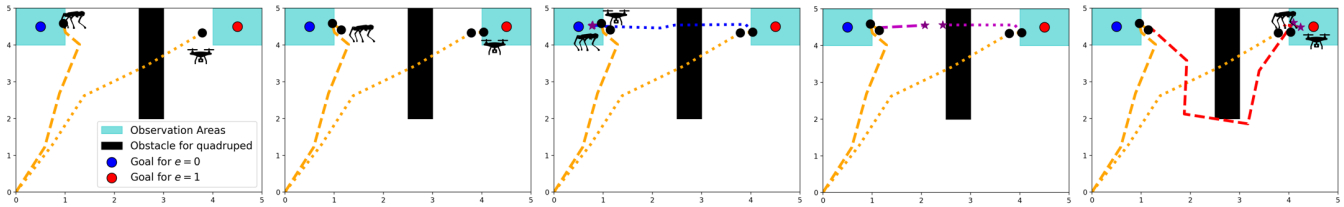


(a) o_0 **(b)** $o_0 = 0$. **(c)** $o_0 = 1$. **(d)** $o_0 = o_1 = 0$.

(e) $o_0 = 0 \neq o_1$. **(f)** $o_0 = 1 \neq o_1$. **(g)** $o_0 = o_1 = 1$. **(h)** Full plan.

Fig. 5: Plan for a quadruped, initiated below the dark obastacle. $e = 0$ and $e = 1$ correspond to the target being located at the blue and red goal nodes, respectively. 80% accurate observations are made in the blue areas.

[1] https://github.com/kasperjo/MixedObservableRRT.git



(a) First observation o_0. **(b)** Second observation o_1. **(c)** $o_0 = o_1 = 0$ **(d)** $o_0 \neq o_1$ **(e)** $o_0 = o_1 = 1$

Fig. 6: Mission plan for a quadruped and a drone. The agents explore one goal node each and take one of three paths depending on the observations.

given that the first observation is $o_0 = 0$, the quadruped follows the blue and purple paths for $o_1 = 0$ and $o_1 = 1$, respectively. Figures 5f and 5g show the corresponding plan, given that the first observation is $o_0 = 1$. The full plan with all seven branches is shown in Figure 5h.

B. Multi-Agent Simulation

This system consists of two robots: a quadruped that can not traverse the dark region, and a drone that can fly over the dark region. There is a science sample in one of the top corners, and the goal is for both robots to locate it. Observations that are correct with 80% probability are made inside the blue regions and communicated between agents.

The agents split up to explore one potential target each (Figure 6a), and the quadruped makes the first observation o_0 in the top left. Regardless of if $o_0 = 0$ or $o_0 = 1$, the drone continues to the top right to make another observation o_1, and the quadruped changes direction; which is fascinating, as even if the quadruped observes $o_0 = 0$, the drone does not change direction, but makes a second observation to account for the possibility of the first observation being wrong. If both observations correspond to $e = 0$, the agents move to the top left (Figure 6c), and if both observations correspond to $e = 1$, the agents move to the top right (Figure 6e). If the observations are different, the agents end up in-between (Figure 6d), since the belief is then back to $[0.5, 0.5]$.

C. Hardware Experiment

We show the result from an experiment on hardware where we consider the exploration mission from Section IV-B, but without noise.

1) Experimental Setup and Hardware Models: The setup is illustrated in Figure 1. A quadruped and a drone are initiated in the origin of a 5m×5m environment, and the mission objective is to locate a science sample.

We model the quadruped and drone as unicycles [25] and leverage two MPC algorithms to make the agents follow way-points from the high-level MORRT plan. At 60Hz the MPC algorithms solve for velocity and yaw-rate inputs, which are fed to the quadruped and drone.

2) Experimental Results: The experiment is illustrated in Figure 7. The quadruped observes $o_0 = 1$ in the second snapshot, and communicates to the drone. Since the observation is perfect, both agents move to the top right goal without making another observation. The drone arrives first and waits for the quadruped to arrive. Note the difference from the simulation in Section IV-B, where a second observation was made to account for the noise.

V. Conclusion

In this work we introduced the MORRT, which plans a mission for heterogeneous multi-agent exploration. Using MORRTs we presented a dynamic program for finding the best mission plan, which showed how autonomy can be enabled in a multi-agent scenario where agents move and act simultaneously. The approach was illustrated, both in simulation and on hardware experiments. It was found that two agents (a quadruped and a drone) made intelligent collaborative decisions based on their observations from the environment. Future works include extending the MORRT approach to account for system dynamics. Another interesting direction is to consider limitations in the communication between agents.

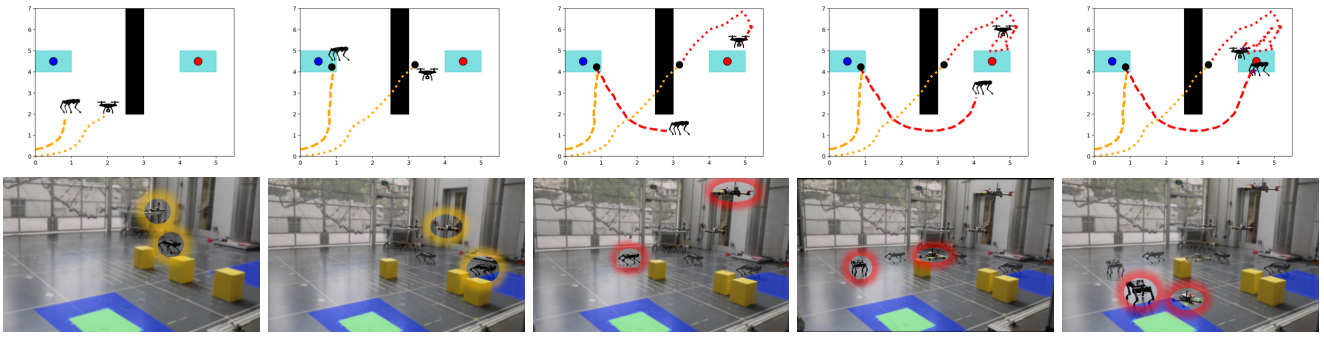


Fig. 7: Experimental results from the setup in Figure 1.

REFERENCES

[1] P. Nilsson, S. Haesaert, R. Thakker, K. Otsu, C. Vasile, A. akbar Aghamohammadi, R. Murray, and A. Ames, "Toward specification-guided active mars exploration for cooperative robot teams," in *Robotics: Science and Systems*, 2018.

[2] G. Lakemeyer and B. Nebel, Eds., *Exploring Artificial Intelligence in the New Millennium*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003.

[3] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, Mass.: MIT Press.

[4] C. Stachniss, *Robotic Mapping and Exploration*. Berlin: Springer, 2009.

[5] U. Rosolia, Y. Chen, S. Daftry, M. Ono, Y. Yue, and A. D. Ames, "The mixed-observable constrained linear quadratic regulator problem: the exact solution and practical algorithms," 2021.

[6] P. J. Goulart, E. C. Kerrigan, and J. M. Maciejowski, "Optimization over state feedback policies for robust control with constraints," *Automatica*, vol. 42, no. 4, pp. 523–533, 2006.

[7] L. Chisci, J. Rossiter, and G. Zappa, "Systems with persistent disturbances: predictive control with restricted constraints," *Automatica*, vol. 37, no. 7, pp. 1019–1028, 2001.

[8] D. Mayne, M. Seron, and S. Raković, "Robust model predictive control of constrained linear systems with bounded disturbances," *Automatica*, vol. 41, no. 2, pp. 219–224, 2005.

[9] S. Yu, C. Maier, H. Chen, and F. Allgöwer, "Tube MPC scheme based on robust control invariant set with application to Lipschitz nonlinear systems," *Systems & Control Letters*, vol. 62, no. 2, pp. 194–200, 2013.

[10] J. Fleming, B. Kouvaritakis, and M. Cannon, "Robust tube MPC for linear systems with multiplicative uncertainty," *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 1087–1092, 2015.

[11] Y.-S. Wang, N. Matni, and J. C. Doyle, "A system-level approach to controller synthesis," *IEEE Transactions on Automatic Control*, vol. 64, no. 10, pp. 4079–4093, 2019.

[12] A. Liniger, X. Zhang, P. Aeschbach, A. Georghiou, and J. Lygeros, "Racing miniature cars: Enhancing performance using stochastic MPC and disturbance feedback," in *2017 American Control Conference (ACC)*, 2017, pp. 5642–5647.

[13] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski, "Adjustable robust solutions of uncertain linear programs1," *Mathematical Programming*, vol. 99, pp. 351–376, 2004.

[14] D. Mayne, S. Raković, R. Findeisen, and F. Allgöwer, "Robust output feedback model predictive control of constrained linear systems," *Automatica*, vol. 42, no. 7, pp. 1217–1222, 2006.

[15] I. Alvarado, D. Limon, T. Alamo, and E. Camacho, "Output feedback robust tube based MPC for tracking of piece-wise constant references," in *2007 46th IEEE Conference on Decision and Control*, 2007, pp. 2175–2180.

[16] M. Cannon, Q. Cheng, B. Kouvaritakis, and S. V. Raković, "Stochastic tube MPC with state estimation," *Automatica*, vol. 48, no. 3, pp. 536–541, 2012.

[17] S. LaValle and J. Kuffner, "Randomized kinodynamic planning." *I. J. Robotic Res.*, vol. 20, pp. 378–400, 2001.

[18] C. Fulgenzi, C. Tay, A. Spalanzani, and C. Laugier, "Probabilistic navigation in dynamic environment using rapidly-exploring random trees and gaussian processes," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 1056–1062.

[19] J. Kim and J. Ostrowski, "Motion planning a aerial robot using rapidly-exploring random trees with dynamic constraints," in *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 2, 2003, pp. 2200–2205 vol.2.

[20] M. Kothari and I. Postlethwaite, "A probabilistically robust path planning algorithm for UAVs using rapidly-exploring random trees," *Journal of Intelligent & Robotic Systems*, vol. 71, 2013.

[21] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," *CoRR*, vol. abs/1005.0416, 2010. [Online]. Available: http://arxiv.org/abs/1005.0416

[22] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the rrt*," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 1478–1483.

[23] V. Krishnamurthy, *Partially Observed Markov Decision Processes: From Filtering to Controlled Sensing*. Cambridge University Press, 2016.

[24] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[25] M. Aicardi, G. Casalino, A. Bicchi, and A. Balestrino, "Closed loop steering of unicycle like vehicles via Lyapunov techniques," *IEEE Robotics Automation Magazine*, vol. 2, no. 1, pp. 27–35, 1995.